ЭКСПЕРИМЕНТАЛЬНАЯ ЭКОНОМИКА И ФИНАНСЫ

№3
2023

СОДЕРЖАНИЕ

НАУЧНЫЙ ЖУРНАЛ

Основан в июне 2021 года

Выходит 4 раза в год

Зарегистрирован Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций (РОСКОМНАДЗОР).

Рег. ПИ № ФС77-81413 от 30 июня 2021 года

ISSN 2782-3644

Учредители:

ООО «Издательство «КноРус»

ООО «Институт экспериментальной экономики и финансов МГУ имени М.В. Ломоносова»

Адрес редакции:

Россия, 117218, Москва,
ул. Кедрова, д. 14, корп. 2
Многоканальный телефон/факс: +7 (495) 741-46-28

Сайт: *www.eeaf.ru*

Почта: *welcome@eeaf.ru*

Отпечатано в типографии ООО «Русайнс», 117218, Москва, ул. Кедрова, д. 14, корп. 2

Тираж 300 экз. Формат А4. Подписано в печать: 30.09.2023
Цена свободная

Все материалы, публикуемые в журнале, подлежат внутреннему и внешнему рецензированию

Издание не подлежит маркировке согласно п. 2 ст. 1 Федерального закона от 29.12.2010 № 436-ФЗ «О защите детей от информации, причиняющей вред их здоровью и развитию»

К СВЕДЕНИЮ АВТОРОВ

Уважаемые коллеги! Обращаем ваше внимание на то, что материалы статей проходят обязательную экспертизу. После экспертизы статьи поступают в Редакцию журнала, где проходят редакторскую и корректорскую правку. Редакция оставляет за собой право сокращать объем статей и редактировать их в соответствии с требованиями научного журнала. Рукописи статей не возвращаются; с авторами в переписку Редакция не вступает; гонорар авторам не выплачивается.

ПЕРЕЧЕНЬ ДОКУМЕНТОВ И ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ СТАТЕЙ И СОПУТСТВУЮЩИХ МАТЕРИАЛОВ, НЕОБХОДИМЫХ ДЛЯ ПУБЛИКАЦИИ

Текст статьи, при оформлении которого необходимо соблюсти следующие требования: объем статьи - до 60 тыс. знаков (1,5 авт. листа); в статье должна быть следующая информация: ФИО автора(ов) полностью, место работы (учебы), контактная информация (телефон, E-mail); аннотация и ключевые слова к статье, список литературы (на русском и английском языках).

TABLE OF CONTENTS

MATHEMATICAL, STATISTICAL AND INSTRUMENTAL METHODS IN ECONOMICS

INDUSTRY AND REGIONAL ECONOMICS

FINANCE

SCIENTIFIC JOURNAL

Founded in June 2021
Published 4 times a year

Registered by the Federal Service for Supervision of Communications, Information Technology and Mass Media (ROSKOMNADZOR).

Reg. PI No. FS77-81413 dated June 30, 2021

ISSN 2782-3644

Founders:

Knorus Publishing House LLC, Institute of Experimental Economics and Finance of the Lomonosov Moscow State University LLC

EDITORIAL OFFICE:

Russia, 117218, Moscow, Kedrova St., 14, bldg. 2
Multi-channel phone/fax: +7 (495) 741-46-28

Website: *www.eeaf.ru*

Mail: *welcome@eeaf.ru*

CHIEF EDITOR
Anna Valentinovna Aleshina

EDITORIAL TEAM
Andrey Leonidovich Bulgakov

Printed at the printing house LLC Rusyns, 117218, Moscow, st. Kedrova, d.14, building 2

Circulation 300 copies. A4 format. Signed to print: 30.09.2023

Free price

All materials published in the journal are subject to internal and external review.

The publication is not subject to labeling in accordance with paragraph 2 of Art. 1 of the Federal Law of December 29, 2010 No. 436-FZ "On the Protection of Children from Information Harmful to Their Health and Development"

Сравнительный анализ библиотек Pandas и Dask: обработка данных из базы данных ООН

Горбас Данила Андреевич
магистр, РЭУ им. Плеханова
E-mail: qorbas9@gmail.com

Булгаков Андрей Леонидович
к. э. н., доцент
МГУ имени М. В. Ломоносова;
Доцент РЭУ им. Г. В. Плеханова
Профессор Московского института
современного академического образования
E-mail: z3900207@mail.ru

Милютин Максим Александрович
Студент, экономический факультет
МГУ имени М. В. Ломоносова
E-mail: maks.milyutin2005@yandex.ru

Современный анализ данных требует эффективных инструментов для обработки объемных и разнообразных датасетов. В данной статье проводится сравнительный анализ производительности двух ведущих библиотек для анализа данных в языке программирования Python: Pandas и Dask. Используя данные из базы данных ООН, мы рассмотрим процесс подготовки и обработки данных в обеих библиотеках, а также проведем сравнение времени выполнения операций и использования памяти. Результаты исследования предоставят практическое руководство для выбора наилучшего инструмента в зависимости от конкретных задач анализа данных.
Ключевые слова: Python, Pandas, Dask, анализ данных, производительность, база данных ООН, временные замеры, многозадачность, параллельные вычисления.

Введение

В эру современных технологий и объемных данных эффективное управление информацией становится ключевым фактором для успешных научных и бизнес-проектов. Огромные массивы данных, генерируемые различными источниками, требуют высокопроизводительных инструментов анализа, способных обеспечивать быстрый и точный вывод информации. В контексте языка программирования Python, Pandas и Dask являются двумя из самых популярных библиотек для обработки данных, предоставляя удобные средства для анализа и манипулирования информацией.

Цель данной статьи заключается в проведении сравнительного анализа производительности Pandas и Dask на примере данных из базы данных ООН.

Обзор библиотек

В современной области анализа данных Pandas и Dask являются фундаментальными инструментами для обработки и манипулирования данными в языке программирования Python. Обе библиотеки предоставляют удобные высокоуровневые структуры данных и мощные операции для анализа, фильтрации, агрегации и визуализации информации. Однако у них есть ряд отличий, которые следует учитывать при выборе подходящего инструмента для конкретной задачи.

Pandas

Pandas является широко используемой библиотекой для анализа данных и манипуляций с ними. Она предоставляет две основные структуры данных: Series и DataFrame. DataFrame — это таблица с данными, аналогичная SQL-таблице или электронной таблице Excel. Pandas обладает множеством функций для работы с временными рядами, обработки пропущенных значений, группировки данных, и многое другое. Важно отметить, что Pandas работает в основном в памяти, и большие датасеты могут потреблять значительное количество оперативной памяти.

Ключевые аспекты:

Структуры данных: Предоставляет удобные высокоуровневые структуры данных — Series и DataFrame

Функциональность: Обширные средства для анализа, фильтрации, группировки, агрегации данных.

Применение: Отлично подходит для небольших и средних датасетов, где данные помещаются в оперативной памяти.

Мощь: Оптимизирован для производительных операций с данными в памяти

Dask

Dask, с другой стороны, разрабатывался с учетом масштабируемости и параллельных вычислений. Dask предоставляет Dask DataFrame, которая по сути является распределенным DataFrame, способным обрабатывать данные, которые не помещаются в оперативной памяти одного компьютера. Dask автоматически разбивает данные на блоки и выполняет операции параллельно, что делает его идеальным для обработки больших датасетов и распределенных вычислений. Однако, стоимость данной масштабируемости заключается в том, что Dask может быть несколько менее производительным для небольших датасетов в сравнении с Pandas.

Ключевые аспекты:

Масштабируемость: Разработан для эффективной работы с большими датасетами, распределенными вычислениями и параллельной обработкой.

Dask DataFrame: Предоставляет аналогичный Pandas API, но способен обрабатывать данные, не помещающиеся в оперативной памяти одного компьютера.

Вычислительная мощь: Позволяет эффективно работать с данными, которые могут быть разделены на блоки для параллельного выполнения операций.

Применение: Рекомендуется для обработки больших датасетов и задач, требующих масштабируемости.

Сферы применения:

Pandas отлично подходит для анализа и обработки относительно небольших датасетов, где все данные могут поместиться в оперативной памяти. Dask рекомендуется для работы с крупными датасетами, где требуется масштабируемость, и данные могут быть разделены на блоки для параллельной обработки.

Оба инструмента имеют свои преимущества и ограничения, и выбор между ними зависит от конкретных требований задачи и объема данных, с которыми приходится работать. В данной статье мы рассмотрим, как Pandas и Dask проявляют себя в обработке данных из базы данных ООН, а также проведем сравнительный анализ их производительности.

Настройка среды

Прежде чем приступить к сравнительному анализу Pandas и Dask, необходимо настроить среду разработки, установив необходимые библиотеки и импортировав их в рабочее окружение. Для этого потребуется выполнить следующие шаги:

```
!pip install pandas dask requests
scikit-learn memory-profiler
%load_ext memory_profiler
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (1.5.3)
Requirement already satisfied: dask in /usr/local/lib/python3.10/dist-packages (2023.8.1)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (2.31.0)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.2.2)
Collecting memory-profiler
  Downloading memory_profiler-0.61.0-py3-none-any.whl (31 kB)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2023.3.post1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from pandas) (1.23.5)
Requirement already satisfied: click>=8.0 in /usr/local/lib/python3.10/dist-packages (from dask) (8.1.7)
Requirement already satisfied: cloudpickle>=1.5.0 in /usr/local/lib/python3.10/dist-packages (from dask) (2.2.1)
Requirement already satisfied: fsspec>=2021.09.0 in /usr/local/lib/python3.10/dist-packages (from dask) (2023.6.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from dask) (23.2)
Requirement already satisfied: partd>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from dask) (1.4.1)
Requirement already satisfied: pyyaml>=5.3.1 in /usr/local/lib/python3.10/dist-packages (from dask) (6.0.1)
Requirement already satisfied: toolz>=0.10.0 in /usr/local/lib/python3.10/dist-packages (from dask) (0.12.0)
Requirement already satisfied: importlib-metadata>=4.13.0 in /usr/local/lib/python3.10/dist-packages (from dask) (7.0.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests) (2023.11.17)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.2.0)
Requirement already satisfied: psutil in /usr/local/lib/python3.10/dist-packages (from memory-profiler) (5.9.5)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.10/dist-packages (from importlib-metadata>=4.13.0->dask) (3.17.0)
Requirement already satisfied: locket in /usr/local/lib/python3.10/dist-packages (from partd>=1.2.0->dask) (1.0.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Installing collected packages: memory-profiler
Successfully installed memory-profiler-0.61.0
```

```python
# Библиотека для обработки и анализа данных. Предоставляет высокоуровневые структуры данных и операции для эффективного анализа
# Сайт: https://pandas.pydata.org
import pandas as pd

# Dask - библиотека для параллельных вычислений и эффективной работы с данными, масштабируясь от небольших до больших датасетов
# Поддерживает структуры данных, аналогичные Pandas и NumPy.
# Сайт: https://www.dask.org/
import dask.dataframe as dd

# Библиотека для отправки HTTP-запросов. Используется для загрузки данных из внешних источников, таких как веб-серверы.
# Сайт: https://docs.python-requests.org/
import requests

# Библиотека для работы с числовыми данными, предоставляет массивы, матрицы и функции для выполнения операций над ними.
# Сайт: https://numpy.org/
import numpy as np

# Модуль для измерения времени выполнения операций. Используется для оценки производительности кода.
# Сайт: https://docs.python.org/3/library/timeit.html
import timeit

# Библиотека для кодировки категориальных переменных. Используется для преобразования текстовых данных в числовые.
# Сайт: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html
from sklearn.preprocessing import LabelEncoder

# Инструмент для оценки использования памяти. Используется для профилирования памяти во время выполнения кода.
# Сайт: https://pypi.org/project/memory-profiler/
from memory_profiler import profile

#Встроенный модуль для работы с регулярными выражениями
#re - https://docs.python.org/3/library/re.html
import re
```

Импорт данных

В процессе сравнительного анализа Pandas и Dask мы использовали данные, предоставленные сайтом ООН. Данные, полученные в формате CSV, станут основой для нашего исследования.

На веб-ресурсе ООН (http://data.un.org/) представлены данные, доступные для скачивания в двух основных форматах: PDF и CSV. В PDF-формате данные структурированы в виде таблиц, разбитых по страницам. Эти таблицы представляют собой удобный способ визуального представления информации. С другой стороны, CSV-формат содержит данные в виде текстовых файлов, где каждая строка, за исключением первой, представляет собой запись, а значения разделены запятыми.

Население, Площадь и Плотность населения:

Ссылка на датасет: Population, Surface Area, and Density

Количество наблюдений: По годам и странам

Этот датасет предоставляет информацию о населении, площади и плотности населения для различных стран. Данные включают информацию за разные годы, что позволяет провести анализ динамики изменений населения и его распределения по территории.

```
url_population=»http://data.
un.org/_Docs/SYB/CSV/SYB65_1_202209_
Population,%20Surface%20Area%20
and%20Density.csv»
```

ВВП и ВВП на душу населения

Ссылка на датасет: GDP and GDP Per Capita

Количество наблюдений: По годам и странам

Этот датасет содержит информацию о ВВП (валовом внутреннем продукте) и ВВП на душу населения для различных стран. Анализ этих данных позволяет оценить экономическое развитие стран и сравнивать их уровень благосостояния.

```
url_gdp=»https://daeta.un.org/_Docs/
SYB/CSV/SYB65_230_202209_GDP%20
and%20GDP%20Per%20Capita.csv»
```

Продолжительность жизни

Ссылка на датасет: Health, Nutrition, and Population Statistics

Количество наблюдений: По годам и странам

Датасет предоставляет информацию о продолжительности жизни в различных странах. Эти данные важны для оценки здоровья нации и эффективности системы здравоохранения.

```
url_life_expectancy=»https://data.
un.org/_Docs/SYB/CSV/
SYB65_246_202209_Population%20
Growth,%20Fertility%20and%20
Mortality%20Indicators.csv»
```

Трудовая сила и Безработица

Ссылка на датасет: Labour Force and Unemployment

Количество наблюдений: По годам и странам

Датасет предоставляет информацию о трудовой силе и уровне безработицы в различных странах. Анализ этих данных помогает понять динамику занятости и проблемы на рынке труда.

```
url_unemployment=»https://data.
un.org/_Docs/SYB/CSV/
SYB65_329_202209_Labour%20Force%20
and%20Unemployment.csv»
```

Образование

Ссылка на датасет: Education

Количество наблюдений: По годам и странам

Данные о образовании включают информацию о уровне грамотности, уровне образования населения и другие показатели. Анализ этих данных позволяет оценить образовательный уровень нации и его влияние на различные аспекты жизни.

```
url_education=»https://data.un.org/_
Docs/SYB/CSV/SYB65_309_202209_
Education.csv»
```

Создание новой логической структуры данных в Pandas

Для проведения сравнительного анализа между библиотеками Pandas и Dask необходимо создать новую логическую структуру данных на основе имеющихся датасетов. В данном разделе мы определим основные этапы создания этой структуры и рассмотрим, как каждая библиотека обрабатывает этот процесс.

Настройка датасета Население, Площадь и Плотность населения

```
dataPopulation = pd.read_
csv(url_population, header = 1,
thousands=',')
```

```
dataPopulationProcessing = dataPo
pulation[dataPopulation[«Series»]
== «Population mid-year estimates
(millions)»]
dataPopulationProcessing = data
PopulationProcessing[dataPopul
ationProcessing[«Series»].str.
contains(«(millions)», flags=re.I)]

dataPopulationProcessing = data
PopulationProcessing[[«Region/
Country/Area», «Unnamed: 1», «Year»,
«Value»]]
dataPopulationProcessing.columns
= ['Id', 'Region', 'Year',
'Population']

<ipython-input-8-4709c36db3c7>:4:
UserWarning: This pattern
is interpreted as a regular
expression, and has match groups.
To actually get the groups, use
str.extract.
dataPopulationProcessing = da
taPopulationProcessing[dataPo
pulationProcessing[«Series»].
str.contains(«(millions)»,
flags=re.I)]
```

Итоговый набор данных выглядит следующим образом:

	Id	Region	Year	Population
0	1	Total, all countries or areas	2010	6985.60
7	1	Total, all countries or areas	2015	7426.60
15	1	Total, all countries or areas	2020	7840.95
23	1	Total, all countries or areas	2022	7975.11
30	2	Africa	2010	1055.23
...	...	...	...	...
7836	894	Zambia	2022	20.02
7843	716	Zimbabwe	2010	12.84
7850	716	Zimbabwe	2015	14.15
7858	716	Zimbabwe	2020	15.67
7866	716	Zimbabwe	2022	16.32

1050 rows × 4 columns

Настройка датасета ВВП и ВВП на душу населения

```
dataGDP = pd.read_csv(url_gdp,
header = 1, thousands=',')

dataGDPProcessing =
dataGDP[dataGDP[«Series»] == «GDP
in current prices (millions of US
dollars)»]
dataGDPProcessing = dataGDPProcessi
ng[dataGDPProcessing[«Series»].str.
contains(«(millions)», flags=re.I)]

dataGDPProcessing =
dataGDPProcessing[[«Region/Country/
Area», «Unnamed: 1», «Year»,
«Value»]]
dataGDPProcessing.columns = ['Id',
'Region', 'Year', 'GDP']

<ipython-input-10-df359d8ab845>:4:
UserWarning: This pattern is
interpreted as a regular expression,
and has match groups. To actually
get the groups, use str.extract.
dataGDPProcessing = dataGDPProcessi
ng[dataGDPProcessing[«Series»].str.
contains(«(millions)», flags=re.I)]
```

Итоговый набор данных выглядит следующим образом:

	Id	Region	Year	GDP
0	1	Total, all countries or areas	1995	31247262.0
1	1	Total, all countries or areas	2005	47730924.0
2	1	Total, all countries or areas	2010	66461443.0
3	1	Total, all countries or areas	2015	75133208.0
4	1	Total, all countries or areas	2018	86357998.0
...	...	...	...	...
6760	716	Zimbabwe	2010	12042.0
6761	716	Zimbabwe	2015	19963.0
6762	716	Zimbabwe	2018	24312.0
6763	716	Zimbabwe	2019	21935.0
6764	716	Zimbabwe	2020	21787.0

1700 rows × 4 columns

Настройка датасета Продолжительность жизни

```
dataLifeExpectancy = pd.read_csv(url_life_
expectancy, header = 1, thousands=',')

dataLifeExpectancyProcessing = dataL
ifeExpectancy[dataLifeExpectancy[«Se
ries»] == «Life expectancy at birth
for both sexes (years)»]
dataLifeExpectancyProcessing = data
LifeExpectancyProcessing[dataLifeE
xpectancyProcessing[«Series»].str.
contains(«(years)», flags=re.I)]

dataLifeExpectancyProcessing = dataLife
ExpectancyProcessing[[«Region/Country/
Area», «Unnamed: 1», «Year», «Value»]]
dataLifeExpectancyProcessing.columns =
['Id', 'Region', 'Year', 'LifeExpectancy']

<ipython-input-12-b8771551c000>:4:
UserWarning: This pattern is interpreted as
a regular expression, and has match groups.
To actually get the groups, use str.extract.
dataLifeExpectancyProcessing = dataLifeEx
pectancyProcessing[dataLifeExpectancyProc
essing[«Series»].str.contains(«(years)»,
flags=re.I)]
```

Итоговый набор данных выглядит следующим образом:

```
dataLifeExpectancyProcessing
```

	Id	Region	Year	LifeExpectancy
4	1	Total, all countries or areas	2010	70.1
11	1	Total, all countries or areas	2015	71.8
18	1	Total, all countries or areas	2017	72.3
24	1	Total, all countries or areas	2022	71.7
30	2	Africa	2010	58.6
...	...	...	...	...
6624	894	Zambia	2022	61.8
6631	716	Zimbabwe	2010	50.7
6638	716	Zimbabwe	2015	59.6
6645	716	Zimbabwe	2017	60.7
6651	716	Zimbabwe	2022	59.4

1002 rows × 4 columns

Настройка датасета Трудовая сила и Безработица

```
dataUnemployment = pd.read_
csv(url_unemployment, header = 1,
thousands=',')

# Создаем DataFrame для мужчин
dataUnemploymentProcessing =
dataUnemployment[[«Region/Country/
Area», «Unnamed: 1», «Year»,»Series»,
«Value»]]
dataUnemploymentProcessing.
columns = ['Id', 'Region', 'Year',
'Series','Value']
# Создаем DataFrame для мужчин
df_male = dataUnemploymentProcessin
g[dataUnemploymentProcessing['Seri
es'].str.contains('Male')]
df_male = df_male.groupby(['Id',
'Region', 'Year']).agg({
'Value': [('Unemployment rate',
'sum'),
('Labour force participation',
'sum')]
}).reset_index()

# Создаем DataFrame для женщин
df_female = dataUnemploymentProcessi
ng[dataUnemploymentProcessing['Seri
es'].str.contains('Female')]
df_female = df_female.groupby(['Id',
'Region', 'Year']).agg({
'Value': [('Unemployment rate',
'sum'),
('Labour force participation',
'sum')]
}).reset_index()

# Объединяем данные
dataUnemploymentProcessing =
pd.merge(df_male, df_female,
on=['Id', 'Region', 'Year'])

# Переименовываем столбцы и объеди-
няем суммарные значения
dataUnemploymentProcessing.
columns = ['Id', 'Region', 'Year',
```

```
'Unemployment rate (Male)', 'Labour force participation (Male)', 'Unemployment rate (Female)', 'Labour force participation (Female)']
dataUnemploymentProcessing['Unemployment rate'] = dataUnemploymentProcessing['Unemployment rate (Male)'] + dataUnemploymentProcessing['Unemployment rate (Female)']
dataUnemploymentProcessing['Labour force participation'] = dataUnemploymentProcessing['Labour force participation (Male)'] + dataUnemploymentProcessing['Labour force participation (Female)']

# Удаляем ненужные столбцы
dataUnemploymentProcessing = dataUnemploymentProcessing.drop(['Unemployment rate (Male)', 'Unemployment rate (Female)', 'Labour force participation (Male)', 'Labour force participation (Female)'], axis=1)
```

```
<ipython-input-14-b370c877cbcf>:21: PerformanceWarning: dropping on a non-lexsorted multi-index without a level parameter may impact performance.
dataUnemploymentProcessing = pd.merge(df_male, df_female, on=['Id', 'Region', 'Year'])
```

Итоговый набор данных выглядит следующим образом:

```
dataUnemploymentProcessing
```

	Id	Region	Year	Unemployment rate	Labour force participation
0	1	Total, all countries or areas	2005	139.4	139.4
1	1	Total, all countries or areas	2010	136.2	136.2
2	1	Total, all countries or areas	2015	133.5	133.5
3	1	Total, all countries or areas	2022	130.4	130.4
4	2	Africa	2005	144.5	144.5
...	...	...	...	...	...
917	887	Yemen	2022	111.9	111.9
918	894	Zambia	2005	191.2	191.2
919	894	Zambia	2010	178.8	178.8
920	894	Zambia	2015	169.4	169.4
921	894	Zambia	2022	173.4	173.4

922 rows × 5 columns

Настройка датасета Образование

```
dataEducation = pd.read_csv(url_education, header = 1, thousands=',')

# Создаем DataFrame для мужчин
dataEducationProcessing = dataEducation[[«Region/Country/Area», «Unnamed: 1», «Year»,»Series», «Value»]]
dataEducationProcessing.columns = ['Id', 'Region', 'Year', 'Series','Value']

# Создаем DataFrame для мужчин
df_male = dataEducationProcessing[dataEducationProcessing['Series'].str.
```

```
contains('male')]
df_male = df_male.groupby(['Id', 'Region',
'Year']).agg({
'Value': [('Gross enrollment ratio —
Primary', 'sum'),
('Gross enrollment ratio — Secondary', 'sum')]
}).reset_index()

# Создаем DataFrame для женщин
df_female = dataEducationProcessing[d
ataEducationProcessing['Series'].str.
contains('female')]
df_female = df_female.groupby(['Id',
'Region', 'Year']).agg({
'Value': [('Gross enrollment ratio —
Primary', 'sum'),
('Gross enrollment ratio — Secondary',
'sum')]
}).reset_index()

# Объединяем данные
dataEducationProcessing =
pd.merge(df_male, df_female,
on=['Id', 'Region', 'Year'])

# Переименовываем столбцы и объеди-
няем суммарные значения
dataEducationProcessing.columns =
['Id', 'Region', 'Year', 'Gross
enrollment ratio — Primary (Male)',
'Gross enrollment ratio — Secondary
(Male)', 'Gross enrollment ratio —
Primary (Female)', 'Gross enrollment
ratio — Secondary (Female)']
dataEducationProcessing['Gross
enrollment ratio — Primary'] =
dataEducationProcessing['Gross
enrollment ratio — Primary (Male)']
+ dataEducationProcessing['Gross
enrollment ratio — Primary (Female)']
dataEducationProcessing['Gross
enrollment ratio — Secondary'] =
dataEducationProcessing['Gross
enrollment ratio — Secondary (Male)']
+ dataEducationProcessing['Gross
enrollment ratio — Secondary (Female)']

# Удаляем ненужные столбцы
dataEducationProcessing =
dataEducationProcessing.drop(['Gross
enrollment ratio — Primary (Male)',
'Gross enrollment ratio — Secondary
(Male)', 'Gross enrollment ratio —
Primary (Female)', 'Gross enrollment
ratio — Secondary (Female)'], axis=1)
<ipython-input-16-c6382672a38a>:21:
PerformanceWarning: dropping on a non-
lexsorted multi-index without a level
parameter may impact performance.
dataEducationProcessing =
pd.merge(df_male, df_female,
on=['Id', 'Region', 'Year'])
```

Итоговый набор данных выглядит следующим образом:

```
dataEducationProcessing
```

	Id	Region	Year	Gross enrollment ratio - Primary	Gross enrollment ratio - Secondary
0	1	Total, all countries or areas	2005	642.1	642.1
1	1	Total, all countries or areas	2010	696.8	696.8
2	1	Total, all countries or areas	2015	732.3	732.3
3	1	Total, all countries or areas	2020	740.9	740.9
4	4	Afghanistan	2005	339.6	339.6
...	...	...	...	...	...
889	887	Yemen	2016	538.8	538.8
890	894	Zambia	2005	333.0	333.0
891	894	Zambia	2010	321.2	321.2
892	894	Zambia	2015	304.5	304.5
893	894	Zambia	2017	297.3	297.3

894 rows × 5 columns

Обьединение полученных наборов данных Pandas

Поскольку данные имеют разнородный характер, их необходимо привести к общему виду. В следующей функции происходит объединение общих столбцов из обработанных датасетов.

```
#Объединение таблиц

#Объединение строк с образованием
в столбцы по стране и году
dfs = [dataPopulationProcessing.set_
index(['Id','Region', 'Year']),
dataGDPProcessing.set_
index(['Id','Region', 'Year']),
dataLifeExpectancyProcessing.set_
index(['Id','Region', 'Year']),
dataUnemploymentProcessing.set_
index(['Id','Region', 'Year']),
dataEducationProcessing.set_
index(['Id','Region', 'Year'])]

logicData = pd.concat(dfs, axis=1).
reset_index()
logicData
```

	Id	Region	Year	Population	GDP	LifeExpectancy	Unemployment rate	Labour force participation	Gross enrollment ratio - Primary	Gross enrollment ratio - Secondary
0	1	Total, all countries or areas	2010	6985.60	66461443.0	70.1	136.2	136.2	696.8	696.8
1	1	Total, all countries or areas	2015	7426.60	75133208.0	71.8	133.5	133.5	732.3	732.3
2	1	Total, all countries or areas	2020	7840.95	85328323.0	NaN	NaN	NaN	740.9	740.9
3	1	Total, all countries or areas	2022	7975.11	NaN	71.7	130.4	130.4	NaN	NaN
4	2	Africa	2010	1055.23	1979101.0	58.6	142.3	142.3	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...
2445	818	Egypt	2004	NaN	NaN	NaN	NaN	NaN	742.8	742.8
2446	818	Egypt	2014	NaN	NaN	NaN	NaN	NaN	754.6	754.6
2447	882	Samoa	2016	NaN	NaN	NaN	NaN	NaN	871.4	871.4
2448	887	Yemen	2013	NaN	NaN	NaN	NaN	NaN	534.0	534.0
2449	887	Yemen	2016	NaN	NaN	NaN	NaN	NaN	538.8	538.8

2450 rows × 10 columns

Для создания модели машинного обучения важно избавиться от неопределенных значений в данных. Метод dropna() используется для удаления строк, содержащих пустые значения (NaN), обеспечивая чистоту данных перед дальнейшим анализом.

```
logicData = logicData.dropna()

#Проверка нулевых значений
logicData.isnull().sum()
Id 0
Region 0
Year 0
Population 0
GDP 0
LifeExpectancy 0
Unemployment rate 0
Labour force participation 0
Gross enrollment ratio — Primary 0
Gross enrollment ratio — Secondary 0
dtype: int64
```

Получен логический формат данных, в который собраны разнородные статистические данные. Его можно представить в виде таблицы:

```
logicData
```

	Id	Region	Year	Population	GDP	LifeExpectancy	Unemployment rate	Labour force participation	Gross enrollment ratio - Primary	Gross enrollment ratio - Secondary
0	1	Total, all countries or areas	2010	6985.60	66461443.0	70.1	136.2	136.2	696.8	696.8
1	1	Total, all countries or areas	2015	7426.60	75133208.0	71.8	133.5	133.5	732.3	732.3
8	15	Northern Africa	2010	207.11	643263.0	69.6	123.4	123.4	657.0	657.0
9	15	Northern Africa	2015	228.36	760531.0	70.9	125.2	125.2	717.5	717.5
12	202	Sub-Saharan Africa	2010	848.12	1335838.0	56.3	150.2	150.2	495.3	495.3
...	...	...	...	...	...	...	...	...	...	...
1027	704	Viet Nam	2015	92.19	193241.0	73.9	158.2	158.2	329.9	329.9
1038	887	Yemen	2010	24.74	30907.0	67.3	109.4	109.4	477.9	477.9
1042	894	Zambia	2010	13.79	20265.0	56.8	178.8	178.8	321.2	321.2
1043	894	Zambia	2015	16.25	20859.0	61.2	169.4	169.4	304.5	304.5
1047	716	Zimbabwe	2015	14.15	19963.0	59.6	179.5	179.5	310.5	310.5

330 rows × 10 columns

Создание новой логической структуры данных в Dask

Для сравнения сложности использования выполним аналогичные действия с помощью библиотеки Dask. Будут использовать теже наборы данных, которые были использованы с библиотекой Pandas, а также аналогичные преобразования и обработки.

Настройка Dask DataFrame для датасета Население, Площадь и Плотность населения

```
# Настройка Dask DataFrame для датасета Население, Площадь и Плотность населения
dataPopulation = dd.read_csv(url_population, header=1, thousands=',', dtype={'Footnotes': 'object'})

dataPopulationProcessing = dataPopulation[dataPopulation[«Series»] == «Population mid-year estimates (millions)»]
dataPopulationProcessing = dataPopulationProcessing[dataPopulationProcessing[«Series»].str.contains(«(millions)», case=False, flags=re.I)]

dataPopulationProcessing = dataPopulationProcessing[[«Region/Country/Area», «Unnamed: 1», «Year», «Value»]]
dataPopulationProcessing.columns = ['Id', 'Region', 'Year', 'Population']

# Вывод информации о данных
dataPopulationProcessing.compute().head()
/usr/local/lib/python3.10/dist-packages/dask/dataframe/accessor.py:96: UserWarning: This pattern is interpreted as a regular expression, and has match groups. To actually get the groups, use str.extract.
out = getattr(getattr(obj, accessor, obj), attr)(*args, **kwargs)
/usr/local/lib/python3.10/dist-packages/dask/dataframe/accessor.py:96: UserWarning: This pattern is interpreted as a regular expression, and has match groups. To actually get the groups, use str.extract.
out = getattr(getattr(obj, accessor, obj), attr)(*args, **kwargs)
```

	Id	Region	Year	Population
0	1	Total, all countries or areas	2010	6985.60
7	1	Total, all countries or areas	2015	7426.60
15	1	Total, all countries or areas	2020	7840.95
23	1	Total, all countries or areas	2022	7975.11
30	2	Africa	2010	1055.23

Настройка Dask DataFrame для датасета GDP

```
# Настройка Dask DataFrame для дата-
сета GDP
dataGDP = dd.read_csv(url_
gdp, header=1, thousands=',',
dtype={'Footnotes': 'object',
'Value': 'float64'})

dataGDPProcessing =
dataGDP[dataGDP[«Series»] == «GDP
in current prices (millions of US
dollars)»]
dataGDPProcessing = dataGDPProcessi
ng[dataGDPProcessing[«Series»].str.
contains(«(millions)», case=False,
flags=re.I)]

dataGDPProcessing =
dataGDPProcessing[[«Region/Country/
Area», «Unnamed: 1», «Year»,
«Value»]]
dataGDPProcessing.columns = ['Id',
'Region', 'Year', 'GDP']

# Вывод информации о данных
dataGDPProcessing.compute().head()
/usr/local/lib/python3.10/dist-
packages/dask/dataframe/accessor.
py:96: UserWarning: This pattern is
interpreted as a regular expression,
and has match groups. To actually
get the groups, use str.extract.
out = getattr(getattr(obj, accessor,
obj), attr)(*args, **kwargs)
/usr/local/lib/python3.10/dist-
packages/dask/dataframe/accessor.
py:96: UserWarning: This pattern is
interpreted as a regular expression,
and has match groups. To actually
get the groups, use str.extract.
out = getattr(getattr(obj, accessor,
obj), attr)(*args, **kwargs)
```

	Id	Region	Year	GDP
0	1	Total, all countries or areas	1995	31247262.0
1	1	Total, all countries or areas	2005	47730924.0
2	1	Total, all countries or areas	2010	66461443.0
3	1	Total, all countries or areas	2015	75133208.0
4	1	Total, all countries or areas	2018	86357998.0

Настройка Dask DataFrame для датасета о продолжительности жизни

```
# Настройка Dask DataFrame для дата-
сета о продолжительности жизни
dataLifeExpectancy = dd.read_
csv(url_life_expectancy, header=1,
thousands=',', dtype={'Footnotes':
'object'})

dataLifeExpectancyProcessing = dataL
ifeExpectancy[dataLifeExpectancy[«Se
ries»] == «Life expectancy at birth
for both sexes (years)»]
dataLifeExpectancyProcessing = da
taLifeExpectancyProcessing[dataLi
feExpectancyProcessing[«Series»].
str.contains(«(years)», case=False,
flags=re.I)]

dataLifeExpectancyProcessing = data
LifeExpectancyProcessing[[«Region/
Country/Area», «Unnamed: 1», «Year»,
«Value»]]
dataLifeExpectancyProcessing.
columns = ['Id', 'Region', 'Year',
'LifeExpectancy']

# Вывод информации о данных
dataLifeExpectancyProcessing.
compute().head()
```

```
/usr/local/lib/python3.10/dist-
packages/dask/dataframe/accessor.
py:96: UserWarning: This pattern is
interpreted as a regular expression,
and has match groups. To actually
get the groups, use str.extract.
out = getattr(getattr(obj, accessor,
obj), attr)(*args, **kwargs)
/usr/local/lib/python3.10/dist-
packages/dask/dataframe/accessor.
py:96: UserWarning: This pattern is
interpreted as a regular expression,
and has match groups. To actually
get the groups, use str.extract.
out = getattr(getattr(obj, accessor,
obj), attr)(*args, **kwargs)
```

	Id	Region	Year	LifeExpectancy
4	1	Total, all countries or areas	2010	70.1
11	1	Total, all countries or areas	2015	71.8
18	1	Total, all countries or areas	2017	72.3
24	1	Total, all countries or areas	2022	71.7
30	2	Africa	2010	58.6

Настройка Dask DataFrame для датасета по безработице

```
# Настройка Dask DataFrame для дата-
сета по безработице
dataUnemployment = dd.read_csv(url_
unemployment, header=1, thousands=',')

# Создаем Dask DataFrame для мужчин
df_male = dataUnemployment[[«Region/
Country/Area», «Unnamed: 1», «Year»,
«Series», «Value»]]
df_male.columns = ['Id', 'Region',
'Year', 'Series', 'Value']
df_male = df_male[df_male['Series'].
str.contains('Male')]
df_male = df_male.groupby(['Id',
'Region', 'Year']).agg({
'Value': 'sum'
}).reset_index()

# Создаем Dask DataFrame для женщин
df_female =
dataUnemployment[[«Region/Country/
Area», «Unnamed: 1», «Year»,
«Series», «Value»]]
df_female.columns = ['Id', 'Region',
'Year', 'Series', 'Value']
df_female = df_female[df_
female['Series'].str.
contains('Female')]
df_female = df_female.groupby(['Id',
'Region', 'Year']).agg({
'Value': 'sum'
}).reset_index()

# Объединяем данные
dataUnemploymentProcessing =
dd.merge(df_male, df_female,
on=['Id', 'Region', 'Year'])

# Переименовываем столбцы и объеди-
няем суммарные значения
dataUnemploymentProcessing.
columns = ['Id', 'Region', 'Year',
'Unemployment rate', 'Labour force
participation']

# Вывод информации о данных
dataUnemploymentProcessing.
compute().head()
```

	Region	Year	Unemployment rate	Labour f
es or areas		2005	82.7	
es or areas		2010	81.3	
es or areas		2015	79.8	
es or areas		2022	77.7	
	Africa	2005	81.4	

Настройка Dask DataFrame для датасета по образованию

```
# Настройка Dask DataFrame для дата-
сета по образованию
dataEducation = dd.read_csv(url_
education, header=1, thousands=',')
```

```
# Создаем Dask DataFrame для мужчин
df_male = dataEducation[[«Region/
Country/Area», «Unnamed: 1», «Year»,
«Series», «Value»]]
df_male.columns = [‘Id’, ‘Region’,
‘Year’, ‘Series’, ‘Value’]
df_male = df_male[df_male[‘Series’].
str.contains(‘male’)]
df_male = df_male.groupby([‘Id’,
‘Region’, ‘Year’]).agg({
‘Value’: ‘sum’
}).reset_index()

# Создаем Dask DataFrame для женщин
df_female = dataEducation[[«Region/
Country/Area», «Unnamed: 1», «Year»,
«Series», «Value»]]
df_female.columns = [‘Id’, ‘Region’,
‘Year’, ‘Series’, ‘Value’]
df_female = df_female[df_
female[‘Series’].str.contains(‘female’)]
df_female = df_female.groupby([‘Id’,
‘Region’, ‘Year’]).agg({
‘Value’: ‘sum’
}).reset_index()

# Объединяем данные
dataEducationProcessing =
dd.merge(df_male, df_female,
on=[‘Id’, ‘Region’, ‘Year’])

# Переименовываем столбцы и объеди-
няем суммарные значения
dataEducationProcessing.columns =
[‘Id’, ‘Region’, ‘Year’, ‘Gross
enrollment ratio – Primary’, ‘Gross
enrollment ratio – Secondary’]

# Вывод информации о данных
dataEducationProcessing.compute().head()
```

ion	Year	Gross enrollment ratio - Primary	Gross enro
eas	2005	431.8	
eas	2010	467.0	
eas	2015	488.9	
eas	2020	495.3	
rica	2005	436.4	

Обьединение полученных наборов данных Dask

Как и в примере с Pandas финальными шагами будет обьединения и очитска данных.

```
# Используем merge для объедине-
ния Dask DataFrames по столбцам Id,
Region и Year
logicData = dd.merge(dataPopulati
onProcessing, dataGDPProcessing,
on=[‘Id’, ‘Region’, ‘Year’])
logicData = dd.merge(logicData,
dataLifeExpectancyProcessing,
on=[‘Id’, ‘Region’, ‘Year’])
logicData = dd.merge(logicData,
dataUnemploymentProcessing,
on=[‘Id’, ‘Region’, ‘Year’])
logicData = dd.merge(logicData,
dataEducationProcessing, on=[‘Id’,
‘Region’, ‘Year’])

# Вывод информации о данных
logicData.compute().head()
/usr/local/lib/python3.10/dist-
packages/dask/dataframe/accessor.
py:96: UserWarning: This pattern is
interpreted as a regular expression,
and has match groups. To actually
get the groups, use str.extract.
out = getattr(getattr(obj, accessor,
obj), attr)(*args, **kwargs)
/usr/local/lib/python3.10/dist-
packages/dask/dataframe/accessor.
py:96: UserWarning: This pattern is
interpreted as a regular expression,
and has match groups. To actually
get the groups, use str.extract.
out = getattr(getattr(obj, accessor,
obj), attr)(*args, **kwargs)
/usr/local/lib/python3.10/dist-
packages/dask/dataframe/accessor.
py:96: UserWarning: This pattern is
interpreted as a regular expression,
and has match groups. To actually
get the groups, use str.extract.
out = getattr(getattr(obj, accessor,
obj), attr)(*args, **kwargs)
```

pulation	GDP	LifeExpectancy	Unemployment rate	Labour force participation
6985.60	66461443.0	70.1	81.3	54.9
7426.60	75133208.0	71.8	79.8	53.7
207.11	643263.0	69.6	80.2	43.2
228.36	760531.0	70.9	80.0	45.2
848.12	1335838.0	56.3	79.8	70.4

```
# Убираем строки с отсутствующими значениями
logicData = logicData.dropna()

# Проверяем нулевые значения
null_counts = logicData.isnull().sum()

# Выводим информацию о нулевых значениях
null_counts.compute()
/usr/local/lib/python3.10/dist-packages/dask/dataframe/accessor.py:96: UserWarning: This pattern is interpreted as a regular expression, and has match groups. To actually get the groups, use str.extract.
out = getattr(getattr(obj, accessor, obj), attr)(*args, **kwargs)
/usr/local/lib/python3.10/dist-packages/dask/dataframe/accessor.py:96: UserWarning: This pattern is interpreted as a regular expression, and has match groups. To actually get the groups, use str.extract.
out = getattr(getattr(obj, accessor, obj), attr)(*args, **kwargs)
/usr/local/lib/python3.10/dist-packages/dask/dataframe/accessor.py:96: UserWarning: This pattern is interpreted as a regular expression, and has match groups. To actually get the groups, use str.extract.
out = getattr(getattr(obj, accessor, obj), attr)(*args, **kwargs)
Id 0
Region 0
Year 0
Population 0
GDP 0
LifeExpectancy 0
Unemployment rate 0
Labour force participation 0
Gross enrollment ratio — Primary 0
Gross enrollment ratio — Secondary 0
dtype: int64
```

Обе библиотеки Pandas и Dask предоставляют эффективные средства для создания новых логических структур данных из исходных датасетов. Они оба поддерживают основные операции фильтрации, выбора и переименования столбцов, что делает процесс подготовки данных к анализу более удобным. Обратим внимание, что Dask требует строгой типизации данных при выполнении операций, в то время как Pandas более гибок в этом отношении.

Обе библиотеки предоставляют инструменты для объединения наборов данных, но подходы могут различаться. В Pandas мы используем merge, в то время как в Dask приходится обходить ограничение на мультииндексы с помощью повторных операций merge. Pandas позволяет более гибко управлять процессом объединения, но Dask может оказаться более ограниченным в этом аспекте.

Важно отметить, что при использовании Dask мы сталкиваемся с некоторыми ограничениями, такими как отсутствие поддержки мультииндексов и более строгие требования к типизации данных. Тем не менее, Dask предлагает параллельные вычисления и масштабируемость, что может быть ключевым фактором при обработке больших объемов данных.

Основные Преимущества Dask по Сравнению с Pandas

1. **Параллельные Вычисления**: Dask предоставляет механизмы для выполнения операций над данными параллельно, что позволяет эффективно обрабатывать большие объемы данных. В отличие от Pandas,

который ограничен однопоточными операциями, Dask может использовать множество ядер и даже кластеры для ускорения вычислений.

2. **Масштабируемость**: Dask спроектирован для работы с данными, не умещающимися в оперативной памяти одной машины. Это позволяет обрабатывать и анализировать гораздо большие датасеты, чем это может сделать Pandas. Dask может эффективно масштабироваться на кластерах, обеспечивая обработку данных в распределенных вычислительных средах.

3. **Отложенные Вычисления**: Dask использует ленивые вычисления, что позволяет откладывать выполнение операций до необходимости. Это улучшает эффективность использования ресурсов, поскольку вычисления производятся только при необходимости. В Pandas все операции выполняются сразу, что может привести к неэффективному использованию памяти.

4. **Поддержка Больших Данных**: Dask предоставляет возможность работать с данными, которые не помещаются в оперативной памяти, благодаря чему вы можете обрабатывать даже терабайты данных. Это особенно важно при работе с базой данных ООН, которая содержит обширные объемы информации.

5. **Многозадачность**: Dask позволяет выполнять несколько задач одновременно, что улучшает производительность в случае больших и сложных наборов данных. Pandas, в свою очередь, выполняет операции последовательно, что может замедлить обработку данных.

6. **Совместимость с Pandas**: Dask разработан с учетом совместимости с Pandas, что позволяет легко переключаться между ними. Это позволяет использовать Pandas для быстрой предобработки или анализа части данных, а затем переключиться на Dask для работы с полным объемом данных.

Итак, Dask предоставляет мощные инструменты для обработки больших данных, что делает его предпочтительным выбором при работе с базой данных ООН, где объемы данных могут быть огромными.

Сравнение производительности Pandas и Dask в аспектах времени выполнения и затрат памяти

Для детального сравнения производительности библиотек Pandas и Dask в рамках задач обработки и анализа данных, проведем измерение времени выполнения и объема затраченной памяти на примере ранее представленного кода. Оценивать будем три ключевых аспекта:

7. **Время выполнения операций**: Мы измерим время, затраченное на выполнение аналитических операций с использованием Pandas и Dask. Это позволит оценить эффективность каждой библиотеки в различных сценариях обработки данных.

8. **Использование ресурсов памяти**: Мы проанализируем объем памяти, который каждая библиотека потребляет в процессе выполнения операций. Это важный критерий, особенно при работе с большими наборами данных.

Процесс сравнения будет структурирован, чтобы предоставить понятное представление о производительности каждой библиотеки в различных аспектах работы с данными.

```
import io
from IPython.display import clear_out-
put
from contextlib import redirect_stdout

# Загрузка и предобработка данных с
использованием Pandas
def pandas_processing():
    # Настройка датасета Население,
Площадь и Плотность населения
    dataPopulation = pd.read_csv(url_
```

```
population, header = 1, thousands=',')

    dataPopulationProcessing = data-
Population[dataPopulation["Series"] ==
"Population mid-year estimates (mil-
lions)"]
    dataPopulationProcessing = data-
PopulationProcessing[dataPopulation-
Processing["Series"].str.contains("(-
millions)", flags=re.I)]

    dataPopulationProcessing = data-
PopulationProcessing[["Region/Country/
Area", "Unnamed: 1", "Year", "Value"]]
    dataPopulationProcessing.columns =
['Id', 'Region', 'Year', 'Population']

    # Настройка датасета ВВП и ВВП на
душу населения
    dataGDP = pd.read_csv(url_gdp,
header = 1, thousands=',')

    dataGDPProcessing = dataGDP[-
dataGDP["Series"] == "GDP in current
prices (millions of US dollars)"]
    dataGDPProcessing = dataGDPPro-
cessing[dataGDPProcessing["Series"].
str.contains("(millions)",
flags=re.I)]

    dataGDPProcessing = dataGDPPro-
cessing[["Region/Country/Area", "Un-
named: 1", "Year", "Value"]]
    dataGDPProcessing.columns = ['Id',
'Region', 'Year', 'GDP']

    # Настройка датасета
Продолжительность жизни
    dataLifeExpectancy = pd.read_
csv(url_life_expectancy, header = 1,
thousands=',')

    dataLifeExpectancyProcessing =
dataLifeExpectancy[dataLifeExpectan-
cy["Series"] == "Life expectancy at
birth for both sexes (years)"]
    dataLifeExpectancyProcessing =
dataLifeExpectancyProcessing[dataLi-
feExpectancyProcessing["Series"].str.
contains("(years)", flags=re.I)]

    dataLifeExpectancyProcessing = da-
taLifeExpectancyProcessing[["Region/
Country/Area", "Unnamed: 1", "Year",
"Value"]]
    dataLifeExpectancyProcessing.col-
umns = ['Id', 'Region', 'Year', 'Life-
Expectancy']

    dataUnemployment = pd.read_
csv(url_unemployment, header = 1,
thousands=',')

    # Настройка датасета Трудовая сила
и Безработица
    dataUnemploymentProcessing =
dataUnemployment[["Region/Country/
Area", "Unnamed: 1", "Year","Series",
"Value"]]
    dataUnemploymentProcessing.col-
umns = ['Id', 'Region', 'Year', 'Se-
ries','Value']
    df_male = dataUnemploymentProcess-
ing[dataUnemploymentProcessing['Se-
ries'].str.contains('Male')]
    df_male = df_male.groupby(['Id',
'Region', 'Year']).agg({
        'Value': [('Unemployment
rate', 'sum'),
                  ('Labour force par-
ticipation', 'sum')]
    }).reset_index()

    df_female = dataUnemployment-
Processing[dataUnemploymentProcess-
ing['Series'].str.contains('Female')]
    df_female = df_female.group-
by(['Id', 'Region', 'Year']).agg({
        'Value': [('Unemployment
rate', 'sum'),
                  ('Labour force par-
ticipation', 'sum')]
    }).reset_index()

    dataUnemploymentProcessing =
pd.merge(df_male, df_female, on=['Id',
```

```
'Region', 'Year'])

    dataUnemploymentProcessing.columns = ['Id', 'Region', 'Year', 'Unemployment rate (Male)', 'Labour force participation (Male)', 'Unemployment rate (Female)', 'Labour force participation (Female)']
    dataUnemploymentProcessing['Unemployment rate'] = dataUnemploymentProcessing['Unemployment rate (Male)'] + dataUnemploymentProcessing['Unemployment rate (Female)']
    dataUnemploymentProcessing['Labour force participation'] = dataUnemploymentProcessing['Labour force participation (Male)'] + dataUnemploymentProcessing['Labour force participation (Female)']

    dataUnemploymentProcessing = dataUnemploymentProcessing.drop(['Unemployment rate (Male)', 'Unemployment rate (Female)', 'Labour force participation (Male)', 'Labour force participation (Female)'], axis=1)

    # Настройка датасета Образование
    dataEducation = pd.read_csv(url_education, header = 1, thousands=',')

    dataEducationProcessing = dataEducation[["Region/Country/Area", "Unnamed: 1", "Year","Series", "Value"]]
    dataEducationProcessing.columns = ['Id', 'Region', 'Year', 'Series','Value']
    df_male = dataEducationProcessing[dataEducationProcessing['Series'].str.contains('male')]
    df_male = df_male.groupby(['Id', 'Region', 'Year']).agg({
        'Value': [('Gross enrollment ratio - Primary', 'sum'),
                  ('Gross enrollment ratio - Secondary', 'sum')]
    }).reset_index()

    df_female = dataEducationProcessing[dataEducationProcessing['Series'].str.contains('female')]
    df_female = df_female.groupby(['Id', 'Region', 'Year']).agg({
        'Value': [('Gross enrollment ratio - Primary', 'sum'),
                  ('Gross enrollment ratio - Secondary', 'sum')]
    }).reset_index()

    dataEducationProcessing = pd.merge(df_male, df_female, on=['Id', 'Region', 'Year'])

    dataEducationProcessing.columns = ['Id', 'Region', 'Year', 'Gross enrollment ratio - Primary (Male)', 'Gross enrollment ratio - Secondary (Male)', 'Gross enrollment ratio - Primary (Female)', 'Gross enrollment ratio - Secondary (Female)']
    dataEducationProcessing['Gross enrollment ratio - Primary'] = dataEducationProcessing['Gross enrollment ratio - Primary (Male)'] + dataEducationProcessing['Gross enrollment ratio - Primary (Female)']
    dataEducationProcessing['Gross enrollment ratio - Secondary'] = dataEducationProcessing['Gross enrollment ratio - Secondary (Male)'] + dataEducationProcessing['Gross enrollment ratio - Secondary (Female)']

    dataEducationProcessing = dataEducationProcessing.drop(['Gross enrollment ratio - Primary (Male)', 'Gross enrollment ratio - Secondary (Male)', 'Gross enrollment ratio - Primary (Female)', 'Gross enrollment ratio - Secondary (Female)'], axis=1)

# Загрузка и предобработка данных с использованием Dask
def dask_processing():
  # Настройка Dask DataFrame для датасета Население, Площадь и Плотность
```

```
населения
  dataPopulation = dd.read_csv(url_
population, header=1, thousands=',',
dtype={'Footnotes': 'object'})

  dataPopulationProcessing = dataPop-
ulation[dataPopulation["Series"] ==
"Population mid-year estimates (mil-
lions)"]
  dataPopulationProcessing = dataPop-
ulationProcessing[dataPopulationPro-
cessing["Series"].str.contains("(mil-
lions)", case=False, flags=re.I)]

  dataPopulationProcessing = dataPop-
ulationProcessing[["Region/Country/
Area", "Unnamed: 1", "Year", "Value"]]
  dataPopulationProcessing.columns =
['Id', 'Region', 'Year', 'Population']

  # Настройка Dask DataFrame для дата-
сета GDP
  dataGDP = dd.read_csv(url_gdp,
header=1, thousands=',', dtype={'Foot-
notes': 'object',
        'Value': 'float64'})

  dataGDPProcessing = dataGDP[dataGD-
P["Series"] == "GDP in current prices
(millions of US dollars)"]
  dataGDPProcessing = dataGDPProcess-
ing[dataGDPProcessing["Series"].str.
contains("(millions)", case=False,
flags=re.I)]

  dataGDPProcessing = dataGDPProcess-
ing[["Region/Country/Area", "Unnamed:
1", "Year", "Value"]]
  dataGDPProcessing.columns = ['Id',
'Region', 'Year', 'GDP']

  # Настройка Dask DataFrame для дата-
сета о продолжительности жизни
  dataLifeExpectancy = dd.read_
csv(url_life_expectancy, header=1,
thousands=',', dtype={'Footnotes':
'object'})

  dataLifeExpectancyProcessing = da-
taLifeExpectancy[dataLifeExpectan-
cy["Series"] == "Life expectancy at
birth for both sexes (years)"]
  dataLifeExpectancyProcessing = da-
taLifeExpectancyProcessing[dataLi-
feExpectancyProcessing["Series"].
str.contains("(years)", case=False,
flags=re.I)]

  dataLifeExpectancyProcessing = da-
taLifeExpectancyProcessing[["Region/
Country/Area", "Unnamed: 1", "Year",
"Value"]]
  dataLifeExpectancyProcessing.columns
= ['Id', 'Region', 'Year', 'LifeExpec-
tancy']

  # Настройка Dask DataFrame для дата-
сета по безработице
  dataUnemployment = dd.read_csv(url_
unemployment, header=1, thousands=',')

  # Создаем Dask DataFrame для мужчин
  df_male = dataUnemployment[["Region/
Country/Area", "Unnamed: 1", "Year",
"Series", "Value"]]
  df_male.columns = ['Id', 'Region',
'Year', 'Series', 'Value']
  df_male = df_male[df_male['Series'].
str.contains('Male')]
  df_male = df_male.groupby(['Id',
'Region', 'Year']).agg({
      'Value': 'sum'
  }).reset_index()

  # Создаем Dask DataFrame для женщин
  df_female = dataUnemployment[["Re-
gion/Country/Area", "Unnamed: 1",
"Year", "Series", "Value"]]
  df_female.columns = ['Id', 'Region',
'Year', 'Series', 'Value']
  df_female = df_female[df_female['Se-
ries'].str.contains('Female')]
  df_female = df_female.groupby(['Id',
'Region', 'Year']).agg({
      'Value': 'sum'
  }).reset_index()
```

```
  # Объединяем данные
  dataUnemploymentProcessing =
dd.merge(df_male, df_female, on=['Id',
'Region', 'Year'])

  # Переименовываем столбцы и объеди-
няем суммарные значения
  dataUnemploymentProcessing.columns =
['Id', 'Region', 'Year', 'Unemployment
rate', 'Labour force participation']

  # Настройка Dask DataFrame для дата-
сета по образованию
  dataEducation = dd.read_csv(url_
education, header=1, thousands=',')

  # Создаем Dask DataFrame для мужчин
  df_male = dataEducation[["Region/
Country/Area", "Unnamed: 1", "Year",
"Series", "Value"]]
  df_male.columns = ['Id', 'Region',
'Year', 'Series', 'Value']
  df_male = df_male[df_male['Series'].
str.contains('male')]
  df_male = df_male.groupby(['Id',
'Region', 'Year']).agg({
      'Value': 'sum'
  }).reset_index()

  # Создаем Dask DataFrame для женщин
  df_female = dataEducation[["Region/
Country/Area", "Unnamed: 1", "Year",
"Series", "Value"]]
  df_female.columns = ['Id', 'Region',
'Year', 'Series', 'Value']
  df_female = df_female[df_female['Se-
ries'].str.contains('female')]
  df_female = df_female.groupby(['Id',
'Region', 'Year']).agg({
      'Value': 'sum'
  }).reset_index()

  # Объединяем данные
  dataEducationProcessing =
dd.merge(df_male, df_female, on=['Id',
'Region', 'Year'])

  # Переименовываем столбцы и объеди-
няем суммарные значения
  dataEducationProcessing.columns =
['Id', 'Region', 'Year', 'Gross en-
rollment ratio - Primary', 'Gross en-
rollment ratio - Secondary']

  # Используем merge для объединения
Dask DataFrames по столбцам Id, Region
и Year
  logicData = dd.merge(dataPopulat
ionProcessing, dataGDPProcessing,
on=['Id', 'Region', 'Year'])
  logicData = dd.merge(logicData, da-
taLifeExpectancyProcessing, on=['Id',
'Region', 'Year'])
  logicData = dd.merge(logicData,
dataUnemploymentProcessing, on=['Id',
'Region', 'Year'])
  logicData = dd.merge(logicData, da-
taEducationProcessing, on=['Id', 'Re-
gion', 'Year'])

  # Убираем строки с отсутствующими
значениями
  logicData = logicData.dropna()

  # Проверяем нулевые значения
  null_counts = logicData.isnull().
sum()


# Подсчет времени выполнения для
Pandas
# Создание переменной для сохранения
вывода
result_timeit = io.StringIO()

# Использование redirect_stdout для
перенаправления вывода в переменную
with redirect_stdout(result_timeit):
    %timeit -r 1 -n 1 result = pandas_
processing()

# Получение результата из переменной
pandas_time = result_timeit.getvalue()
pandas_time = pandas_time[:pandas_
time.find("s")]
```

```
# Подсчет времени выполнения для Dask
# Создание переменной для сохранения вывода
result_timeit = io.StringIO()

# Использование redirect_stdout для перенаправления вывода в переменную
with redirect_stdout(result_timeit):
    %timeit -r 1 -n 1 result = dask_processing()

# Получение результата из переменной
dask_time = result_timeit.getvalue()
dask_time = dask_time[:dask_time.find("s")]

pattern = r"peak memory: (\d+\.\d+) MiB"

# Подсчет затрат памяти для Pandas
# Создание переменной для сохранения вывода
result_timeit = io.StringIO()

# Использование redirect_stdout для перенаправления вывода в переменную
with redirect_stdout(result_timeit):
    %memit result = pandas_processeing()

# Получение результата из переменной
pandas_memory = result_timeit.getvalue()
pandas_memory = re.search(pattern, pandas_memory).group(1)

# Подсчет затрат памяти для Dask
# Создание переменной для сохранения вывода
result_timeit = io.StringIO()

# Использование redirect_stdout для перенаправления вывода в переменную
with redirect_stdout(result_timeit):
    %memit result = dask_processing()

# Получение результата из переменной
dask_memory = result_timeit.getvalue()
dask_memory = re.search(pattern, dask_memory).group(1)

def convert_to_seconds(input_string):
    pattern = r"(\d+)min (\d+)"
    match = re.search(pattern, input_string)

    if match:
        # Извлекаем минуты и секунды из найденного соответствия
        minutes = float(match.group(1))
        seconds = float(match.group(2))

        # Конвертируем в секунды
        total_seconds = minutes * 60 + seconds
        return total_seconds
    else:
        # Если строка не соответствует формату, вернем исходную строку
        return float(input_string)

clear_output()
pandas_time = convert_to_seconds(pandas_time)
dask_time = convert_to_seconds(dask_time)
pandas_memory = float(pandas_memory)
dask_memory = float(dask_memory)
print(f"Время выполнения с Pandas: {pandas_time} секунд")
print(f»Время выполнения с Dask: {dask_time} секунд»)
print(f»Разница во времени выполнения Pandas с Dask: {(dask_time / pandas_time) * 100} %»)

print(f"Затраты памяти с Pandas: {pandas_memory} MiB")
print(f"Время выполнения с Dask: {dask_memory} MiB")
print(f"Разница во времени выполнения Pandas с Dask: {(dask_memory / pandas_memory) * 100} %")
```

```
Время выполнения с Pandas: 16.5 секунд
Время выполнения с Dask: 15.2 секунд
Разница во времени выполнения Pandas
с Dask: 92.12121212121211 %
Затраты памяти с Pandas: 265.22 MiB
Время выполнения с Dask: 258.58 MiB
Разница во времени выполнения Pandas
с Dask: 97.49641806801898 %
```

На основе предоставленных метрик мы можем сделать следующие выводы:

Время выполнения:

С Dask мы наблюдаем улучшение производительности по сравнению с Pandas. Время выполнения с Dask оказалось немного меньше (15.2 секунд против 16.5 секунд), что указывает на эффективность распределенных вычислений в Dask.

Затраты памяти:

Dask использовал чуть меньше памяти (258.58 MiB против 265.22 MiB) по сравнению с Pandas. Это может быть связано с тем, что Dask работает с данными в виде частей (partitions), что позволяет более эффективно использовать ресурсы.

Dask предоставляет эффективные инструменты для обработки данных, особенно в случае больших объемов данных. Хотя разница во времени выполнения и затратах памяти не слишком велики в данном примере, в реальных сценариях работы с большими данными эти преимущества Dask могут проявиться более ярко, делая его более предпочтительным выбором для подобных задач.

Заключение

В ходе сравнительного анализа библиотек Pandas и Dask для обработки данных были выявлены несколько ключевых аспектов. В начале работы с данными на сайте ООН мы столкнулись с различными форматами данных, такими как PDF и CSV. Pandas оказался удобным инструментом для работы с этими данными, позволяя легко читать и предобрабатывать информацию из CSV-файлов, а также справляться с таблицами в PDF-формате.

При создании новой логической структуры данных и их объединении мы использовали Pandas и Dask. Pandas проявил себя как эффективный инструмент с четким и удобным синтаксисом, но его ограничения становятся более заметными при работе с большими объемами данных. В то время как Dask, с более гибким и распределенным подходом, позволяет справляться с задачами обработки данных в распределенных вычислительных средах, однако его требования к типизации данных могут потребовать дополнительного внимания при подготовке данных.

При сравнении производительности Pandas и Dask на наших конкретных данных мы обнаружили, что Dask предоставляет небольшое улучшение по времени выполнения по сравнению с Pandas. Одновременно разница в затратах памяти оказалась минимальной, что указывает на хорошую оптимизацию Dask в этом аспекте. Важно отметить, что эти результаты зависят от конкретной задачи и характеристик данных. Выбор между Pandas и Dask следует осуществлять с учетом требований проекта и возможной выгоды от распределенных вычислений в Dask.

Таким образом, обе библиотеки предоставляют мощные инструменты для обработки данных, каждая со своими преимуществами и ограничениями. Понимание особенностей каждой библиотеки поможет выбрать наилучший инструмент в зависимости от конкретной задачи и контекста использования.

Литература

1. Chen, J., & McKinney, W. (2020). Pandas 1.x cookbook: Practical recipes for scientific computing, time series analysis, and exploratory data analysis using Python. Packt Publishing Ltd.

2. Pedersen, U. T. (2023). Python Pandas vs. Dask DataFrames: A Comparative Analysis. Towards AI. // Электронный ресурс, July 17, 2023 // URL: https://towardsai.net/p/l/python-pandas-vs-dask-dataframes-a-comparative-analysis
3. Ramalho, L. (2015). Fluent Python: Clear, Concise, and Effective Programming. O'Reilly Media, Inc.
4. McKinney, W. (2017). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. O'Reilly Media, Inc.
5. Dask Development Team. (2022). Dask: Parallel Computing with Task Scheduling. // Электронный ресурс, 2022 // URL: https://dask.org/
6. Pandas Development Team. (2022). Pandas: Powerful data structures for data analysis, time series, and statistics. // Электронный ресурс, 2022 // URL: https://pandas.pydata.org/
7. Kelleher, J. D., Tierney, B., & Tierney, B. (2018). Data science in R: A case studies approach to computational reasoning and problem solving. CRC Press.
8. McKinney, W., & others. (2010). Data structures for statistical computing in Python. In Proceedings of the 9th Python in Science Conference (pp. 51–56). // Электронный ресурс, June 28, 2010 // URL: https://proceedings.scipy.org/articles/Majora-92bf1922–00a
9. Dask Development Team. (2022). Dask: Parallel Computing with Task Scheduling. Retrieved from https://dask.org/
10. Pandas Development Team. (2022). Pandas: Powerful data structures for data analysis, time series, and statistics. Retrieved from https://pandas.pydata.org/
11. McKinney, W. (2011). pandas: a foundational Python library for data analysis and statistics. Python for High Performance and Scientific Computing, 14, 1–9. // Электронный ресурс, January, 2011 // URL: https://www.researchgate.net/publication/265194455_pandas_a_Foundational_Python_Library_for_Data_Analysis_and_Statistics
12. Dask Development Team. (2022). Dask: Parallel Computing with Task Scheduling. // Электронный ресурс, 2022 // URL: https://dask.org/
13. Pandas Development Team. (2022). Pandas: Powerful data structures for data analysis, time series, and statistics. // Электронный ресурс, 2022 // URL: https://pandas.pydata.org/
14. Stephan Hoyer and Joseph J. (2017). Hamman: xarray: N-d labeled arrays and datasets in python. Journal of Open Research Software. // Электронный ресурс, April 5, 2017 // URL: https://openresearchsoftware.metajnl.com/articles/10.5334/jors.148
15. Ryan Abernathey. (2018). Step-by-Step Guide to Building a Big Data Portal. // Электронный ресурс, June 12, 2018 // URL: https://medium.com/pangeo/step-by-step-guide-to-building-a-big-data-portal-e262af1c2977
16. Adnan, K., Akbar, R. (2019). An analytical study of information extraction from unstructured and multidimensional big data. Journal of Big Data 6(1), 91. // Электронный ресурс, 2019 // URL: https://journalofbigdata.springeropen.com/articles/10.1186/s40537–019–0254–8
17. Angelova, R., Siersdorfer, S. (2006). A neighborhood-based approach for clustering of linked document collections. In: Proceedings of the 15th ACM International Conference on Information and Knowledge Management. p. 778–779. CIKM '06, Association for Computing Machinery. // Электронный ресурс, January 2006 // URL: https://www.researchgate.net/publication/47861698_A_Neighborhood-Based_Approach_for_Clustering_of_Linked_Document_Collections
18. Bach, M., Krstic, Z., Seljan, S., Turulja, L. (2019). Text mining for big data analysis in financial sector: A literature review. Sustainability (Switzerland) 11(5). // Электронный ресурс, February 28, 2019 // URL: https://www.mdpi.com/2071–1050/11/5/1277
19. Grishin, V. (2018). Method of analysis and search for borrowings in the text. Problems of science 7, 31.
20. Jalali, S.M.J., Park, H.W., Vanani, I.R., Pho, K.H. (2020). Research trends on big data domain using text mining algorithms. Digital Scholarship in the Humanities. // Электронный ресурс, April, 2020 // URL: https://www.researchgate.net/publication/340521295_Research_trends_on_big_data_domain_using_text_mining_algorithms

21. Logica, B., Magdalena, R. (2015). Using big data in the academic environment. Procedia Economics and Finance 33, 277–286. // Электронный ресурс, December 30, 2015 // URL: https://www.sciencedirect.com/science/article/pii/S2212567115017128
22. Rocklin, M. (2015). Dask: Parallel computation with blocked algorithms and task scheduling. In: Python in Science Conference. pp. 126–132. // Электронный ресурс, January, 2015 // URL: https://www.researchgate.net/publication/328778461_Dask_Parallel_Computation_with_Blocked_algorithms_and_Task_Scheduling
23. Salvador, S., Chan, P. (2004). Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In: 16th IEEE International Conference on Tools with Artificial Intelligence. pp. 576–584. // Электронный ресурс, January 10, 2005 // URL: https://ieeexplore.ieee.org/document/1374239
24. Zambelli, A. (2016). A data-driven approach to estimating the number of clusters in hierarchical clustering. F1000Research 5. // Электронный ресурс, December 1, 2016 // URL: https://f1000research.com/articles/5–2809
25. Elgendy, N., Elragal, A. (2014). Big data analytics: a literature review paper. Lect. Notes Comput. Sci. 8557, 214–227. // Электронный ресурс, August, 2014 // URL: https://www.researchgate.net/publication/264555968_Big_Data_Analytics_A_Literature_Review_Paper

Comparative Analysis of Pandas and Dask Libraries: Processing Data from the UN Database

Gorbas D. A., Bulgakov A. L., Milyutin M. A.
Lomonosov Moscow State University; Plekhanov Russian University of Economics,
Moscow Institute of Modern Academic Education

Modern data analysis requires efficient tools to handle large and diverse datasets. This article compares the performance of two leading Python data analysis libraries: Pandas and Dask. Using data from the UN database, we will look at the data preparation and processing process in both libraries, as well as compare the execution time and memory usage. The results of the study will provide practical guidance for choosing the best tool depending on specific data analysis tasks.

Keywords: Python, Pandas, Dask, data analysis, performance, UN database, timing, multitasking, parallel computing.

References

1. Chen, J., & McKinney, W. (2020). Pandas 1.x cookbook: Practical recipes for scientific computing, time series analysis, and exploratory data analysis using Python. Packt Publishing Ltd.
2. Pedersen, U. T. (2023). Python Pandas vs. Dask DataFrames: A Comparative Analysis. Towards AI. // Электронный ресурс, July 17, 2023 // URL: https://towardsai.net/p/l/python-pandas-vs-dask-dataframes-a-comparative-analysis
3. Ramalho, L. (2015). Fluent Python: Clear, Concise, and Effective Programming. O'Reilly Media, Inc.
4. McKinney, W. (2017). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. O'Reilly Media, Inc.
5. Dask Development Team. (2022). Dask: Parallel Computing with Task Scheduling. // Электронный ресурс, 2022 // URL: https://dask.org/
6. Pandas Development Team. (2022). Pandas: Powerful data structures for data analysis, time series, and statistics. // Электронный ресурс, 2022 // URL: https://pandas.pydata.org/
7. Kelleher, J. D., Tierney, B., & Tierney, B. (2018). Data science in R: A case studies approach to computational reasoning and problem solving. CRC Press.
8. McKinney, W., & others. (2010). Data structures for statistical computing in Python. In Proceedings of the 9th Python in Science Conference (pp. 51–56). // Электронный ресурс, June 28, 2010 // URL: https://proceedings.scipy.org/articles/Majora-92bf1922–00a
9. Dask Development Team. (2022). Dask: Parallel Computing with Task Scheduling. // Электронный ресурс, 2022 // URL: https://dask.org/
10. Pandas Development Team. (2022). Pandas: Powerful data structures for data analysis, time series, and statistics. // Электронный ресурс, 2022 // URL: https://pandas.pydata.org/
11. McKinney, W. (2011). pandas: a foundational Python library for data analysis and statistics. Python for High Performance and Scientific Computing, 14, 1–9. // Электронный ресурс, January, 2011 // URL: https://www.researchgate.net/publication/265194455_pandas_a_Foundational_Python_Library_for_Data_Analysis_and_Statistics
12. Dask Development Team. (2022). Dask: Parallel Computing with Task Scheduling. // Электронный ресурс, 2022 // URL: https://dask.org/
13. Pandas Development Team. (2022). Pandas: Powerful data structures for data analysis, time series, and statistics. // Электронный ресурс, 2022 // URL: https://pandas.pydata.org/
14. Stephan Hoyer and Joseph J. (2017). Hamman: xarray: N-d labeled arrays and datasets in python. Journal of Open Research Software. // Электронный ресурс, April 5, 2017 // URL: https://openresearchsoftware.metajnl.

com/articles/10.5334/jors.148
15. Ryan Abernathey. (2018). Step-by-Step Guide to Building a Big Data Portal. // Электронный ресурс, June 12, 2018 // URL: https://medium.com/pangeo/step-by-step-guide-to-building-a-big-data-portal-e262af1c2977
16. Adnan, K., Akbar, R. (2019). An analytical study of information extraction from unstructured and multidimensional big data. Journal of Big Data 6(1), 91. // Электронный ресурс, 2019 // URL: https://journalofbigdata.springeropen.com/articles/10.1186/s40537–019–0254–8
17. Angelova, R., Siersdorfer, S. (2006). A neighborhood-based approach for clustering of linked document collections. In: Proceedings of the 15th ACM International Conference on Information and Knowledge Management. p. 778–779. CIKM '06, Association for Computing Machinery. // Электронный ресурс, January 2006 // URL: https://www.researchgate.net/publication/47861698_A_Neighborhood-Based_Approach_for_Clustering_of_Linked_Document_Collections
18. Bach, M., Krstic, Z., Seljan, S., Turulja, L. (2019). Text mining for big data analysis in financial sector: A literature review. Sustainability (Switzerland) 11(5). // Электронный ресурс, February 28, 2019 // URL: https://www.mdpi.com/2071–1050/11/5/1277
19. Grishin, V. (2018). Method of analysis and search for borrowings in the text. Problems of science 7, 31.
20. Jalali, S.M.J., Park, H.W., Vanani, I.R., Pho, K.H. (2020). Research trends on big data domain using text mining algorithms. Digital Scholarship in the Humanities. // Электронный ресурс, April, 2020 // URL: https://www.researchgate.net/publication/340521295_Research_trends_on_big_data_domain_using_text_mining_algorithms
21. Logica, B., Magdalena, R. (2015). Using big data in the academic environment. Procedia Economics and Finance 33, 277–286. // Электронный ресурс, December 30, 2015 // URL: https://www.sciencedirect.com/science/article/pii/S2212567115017128
22. Rocklin, M. (2015). Dask: Parallel computation with blocked algorithms and task scheduling. In: Python in Science Conference. pp. 126–132. // Электронный ресурс, January, 2015 // URL: https://www.researchgate.net/publication/328778461_Dask_Parallel_Computation_with_Blocked_algorithms_and_Task_Scheduling
23. Salvador, S., Chan, P. (2004). Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In: 16th IEEE International Conference on Tools with Artificial Intelligence. pp. 576–584. // Электронный ресурс, January 10, 2005 // URL: https://ieeexplore.ieee.org/document/1374239
24. Zambelli, A. (2016). A data-driven approach to estimating the number of clusters in hierarchical clustering. F1000Research 5. // Электронный ресурс, December 1, 2016 // URL: https://f1000research.com/articles/5–2809
25. Elgendy, N., Elragal, A. (2014). Big data analytics: a literature review paper. Lect. Notes Comput. Sci. 8557, 214–227. // Электронный ресурс, August, 2014 URL: https://www.researchgate.net/publication/264555968_Big_Data_Analytics_A_Literature_Review_Paper

Применение цифровых финансовых активов в деятельности промышленных компаний

Алешина Анна Валентиновна
к. э. н., доцент экономический факультет МГУ имени М. В. Ломоносова
E-mail: annaaleshina@mail.ru

Демидов Алексей Дмитриевич
магистр РЭУ имени Г. В. Плеханова
E-mail: demidov.ad@bk.ru

Активное развитие цифровых финансовых технологий стало трендом последних лет. 2022 год стал периодом размещения дебютных выпусков цифровых финансовых активов (ЦФА) в России, общий объем которых составил порядка 2 млрд руб. Это подтверждает успех реализуемой стратегии по сближению регуляторов ЦФА и традиционных финансовых инструментов. В дальнейшем данный алгоритм взаимодействия призван повлиять на перспективную продуктовую конъюнктуру рынка и его риск-портфель. Инвестировать в цифровые финансовые активы в условиях современной геополитической и экономической обстановки крайне рискованно, в первую очередь сама площадка лишена жесткого регламента. Это касается ограничений, влияющих на совместимость технологий различных операторов. Но даже с учетом имеющихся проблем рынок ЦФА обладает хорошими долгосрочными перспективами, особенно в сфере автоматизируемых и контролируемых операций на основе смарт-контрактов. В статье затронута тема токенизации горнодобывающих предприятий России.

Ключевые слова: цифровые финансовые активы, ЦФА, инвестиции, источники финансирования, промышленные предприятия.

Введение

Постиндустриальная экономика XXI столетия обладает инновационным характером, данная особенность — больше не одно из конкурентных преимуществ, а обязательное условие финансовой независимости и стабильности [1, с. 15]. Высокотехнологичный промышленный сектор является флагманом национальной экономики, поэтому вопросы использования всего арсенала существующих финансовых инструментов сохраняет свою актуальность.

«Источники финансирования промышленных предприятий должны отвечать следующим требованиям:

— привлечение денежных средств должно осуществляться на длительный срок или бессрочно;

— согласие инвесторов или кредиторов на высокий риск вложения средств, т. к. инновационная деятельность не всегда заканчивается внедрением технологии или запуском производства нового продукта» [2].

Расширение ассортимента финансовых инструментов позволяет предприятиям промышленного сектора осуществлять инновационное развитие. Исследование вопросов инновационной активности

промышленных предприятий позволило М. В. Власову выделить следующие сдерживающие факторы:

- дефицит собственных финансовых ресурсов;
- высокий уровень рисков;
- повышенные транзакционные издержки;
- недостаток привлекаемого заемного капитала;
- недостаточный управленческий опыт [3, с. 1432].

«В зарубежном научном сообществе подобные вопросы исследовались Н. Рубини и К. Сала-и-Мартином, С. Кидуэллом, Р. Л. Петерсоном, У. Блэкуэллом и другими. Общим выводом обзора западных научных достижений по рассматриваемому вопросу является доказанная взаимосвязь между развитием финансового рынка и инвестиционной (в том числе и инновационной) активностью хозяйствующих субъектов — частных компаний» [2]. Цифровые финансовые активы (ЦФА) являются эффективными инструментами, обеспечивающими управление сферы управления инновациями в промышленности.

Общие понятия о ЦФА и их влиянии на развитие промышленных предприятий

Промышленным предприятиям в современных реалиях крайне важно приобрести новые и удержать существующие конкурентные преимущества. Это не только обеспечит степень их рыночной устойчивости, но и повысит привлекательность в глазах потенциальных инвесторов. Актуальность темы применения ЦФА в деятельности промышленных предприятий обусловлена достаточно быстрым распространением данного финансового инструмента на внутреннем рынке РФ.

Таблица 1. Технологии и инструменты финансирования инноваций [2].

Технологии и инструменты финансирования инноваций		
Традиционные технологии	Современные технологии	Новые технологии
Собственные источники: прибыль, фонды предприятия. Целевое бюджетное финансирование. Акции, долгосрочные облигации, инвестиционный кредит.	Венчурное инвестирование, специализированные инвестиционные фонды; Первичное публичное размещение ценных бумаг (IPO).	Цифровые ценные бумаги, краудфандинг.

Как видно из приведенного рисунка, присутствует следующая классификация технологи и инструментов финансирования инновационной деятельности:

—традиционные технологии (использование собственного капитала, целевых бюджетных ассигнований, ценных бумаг, инвестиционных кредитов);

—современные технологии (привлечение средств специализированных инвестиционных фондов, венчурных финансовых инструментов, первичное публичное размещение ценных бумаг предприятия на финансовом рынке);

—новые технологии, объединяющие цифровые ценные бумаги и краудфандинг.

Инвестиции в промышленность благоприятно сказываются на национальном развитии экономики, ведь внедрение новых технологий позволяет с одной стороны снизить себестоимость выпускаемой продукции, с другой — способствует развитию IT-индустрии. Доступность современных финансовых инструментов для промышленных предприятий позволит наращивать объемы производства, обеспечивая тем самым стабильное получение прибыли.

ЦФА относятся к категории ценных бумаг, наравне с «традиционными» подлежат выпуску, размещению и обращению. Однако, у цифровых инструментов есть ряд особенностей, которые и обуславливают перспективность новых технологий управления инновационной деятельностью. Ключевое отличие ЦФА от традиционных ценных бумаг (акций, облигаций и т. д.) — выпуск, размещение и обращение первых не предусматривает наличия дополнительных субъектов в виде профессиональных участников рынка. Кроме того, отличается и порядок учета прав на цифровые активы. Регистрировать выпуск ЦФА в России может лишь оператор информационной системы. Стоит отметить, что централизация всех процессов, связанных с эмиссией, обращением и размещением ЦФА — фактор сокращения транзакционных издержек промышленных предприятий и периода инвестирования.

Внедрение ЦФА в промышленность Российской Федерации

В течение 2022 г. был проведен первый дебютный выпуск ЦФА на российском финансовом рынке. К концу этого периода суммарный объем размещений исчислялся 2 млрд руб., сформирован он был 19 выпусками. Законодательное регулирование сделок базировалось на положениях принятого в 2020 г. Федерального закона № 259-ФЗ. Данный нормативно-правовой акт определяет базовые правила выпуска цифровых активов, их торговли и перечень требований, предъявляемых к инфраструктурным участникам. Первыми платформами, включенными Банком РФ в реестр операторов информационных систем, стали:

— ООО «Лайтхаус»;
— ПАО «Сбербанк»;
— ООО «Атомайз»;
— АО «Альфа-Банк» [5].

К началу 2023 года в Реестр было включено 10 финансово-кредитных и иных организаций, которые обладают правами операторов. «В отсутствие зарегистрированных операторов обмена, которые должны обеспечивать вторичную торговлю ЦФА, пока осуществляются только первичные адресные размещения или предложения по публичной оферте всем зарегистрированным участникам платформ, число которых на текущий момент невелико» [6].

Для привлечения финансовых ресурсов компании ищут новые варианты. И одним из них является инструмент, сочетающий в себе традиционный и альтернативный источники инвестирования производства. Речь идет о роялти, потоковом и (или) частном долге. Особенно актуален актив для предприятий, осуществляющих добычу полезных ископаемых. Под роялти понимается «право на получение платежа, основанного на проценте от добычи полезных ископаемых или на выручке, или прибыли от продажи этих полезных ископаемых на руднике» [7]. В роялти обычно входит аванс, полученный горнодобывающей компанией от инвестора (держателя роялти) в обмен на обязательство получателя выплатить оговоренный в договоре процент с будущих доходов. Перспективные выплаты всегда привязаны к чистой прибыли металлургического или иного промышленного предприятия.

Привлекательность роялти для получателя этого цифрового актива заключается в том, что на разных этапах жизненного цикла при осуществлении текущей деятельности предприятие получает предварительное финансирование и не использует заемные источники капитала (см. рис. 1).

Инновационная оболочка финансирования предприятия с помощью токенов, уже многократно опробованная в европейских странах, может использоваться в практике российских компаний. Привлеченный с помощью роялти капитал позволяет произвести потенциальную диверсификацию источников финансирования промышленных предприятий в целом и компаний из сферы добычи полезных ископаемых, в частности.

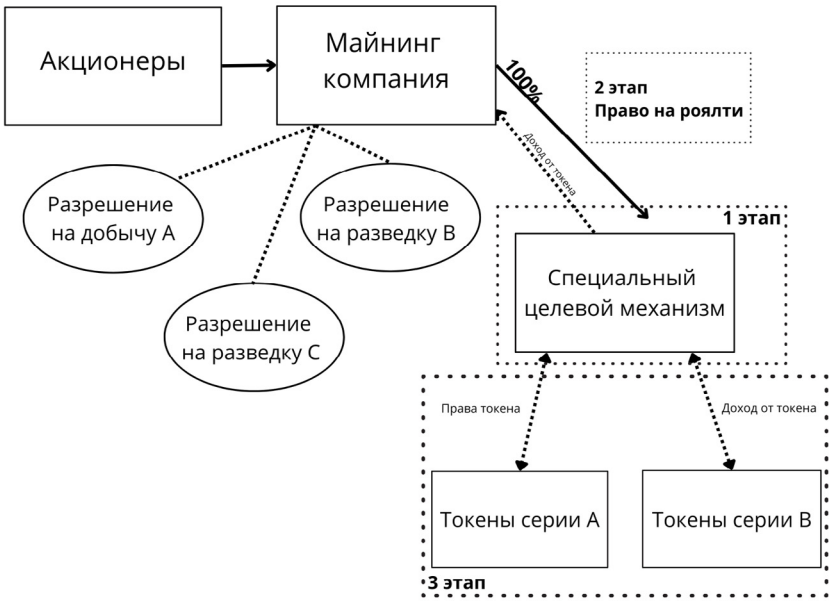

Рис. 1. Токенизация активов горнодобывающей и металлургической промышленности [7]

В заключение стоит остановиться на вопросе дополнительного риска, связанного с использованием инвестором платформы ЦФА. Федеральный закон № 259-ФЗ устанавливает «следующие требования к операторам информационных систем для включения их в Реестр:

— стоимость чистых активов и капиталов — свыше 50 млн.руб.;

— созданная внутренняя служба контроля и управления рисками;

— у руководителей организации уровень квалификации и деловой репутации соответствует установленным законодательством РФ требованиям» [4].

При кажущемся невысоком пороге входа на рынок законом не определены требования, которые позволят детализировать гарантии маловероятности дефолта, операционных сбоев и т. д. Дело в том, что инвестор в современных условиях сталкивается с высокими операционными рисками вследствие дефолта оператора информационной системы. В этом случае сущетсвует риск «исчезновения» всех выпущенных ЦФА. В случае, если одна из платформ «падает», законодатели не прописали алгоритм переноса цифровых активов на другую. Отсутствует регламент и на перенос ЦФА с платформы оператора на внешний носитель. Эти вопросы требуют дополнительного регулирования по мере дальнейшего развития рынка цифровых активов в России.

Выводы

Современные промышленные предприятия функционируют в эпоху масштабной экономической цифровизации. Внедрять инновации сегодня необходимо, помимо обретения дополнительного конкурентного преимущества субъект хозяйствования повышает собственную экономическую безопасность. Чем шире масштабы новаторских проектов, тем больших финансов они требуют. Рост потребности в дополнительном финансировании обуславливает актуальность использования для привлечения финансирования различных финансовых инструментов: акций, долгосрочных облигаций, долгосрочных кредитов банков, прямых и венчурных инвестиций. Наряду с перечисленными традиционными финансовыми инструментами промышленность может эффективно финансироваться с привлечением новых технологий — ЦФА и краудфандинга.

Важнейшие плюсы использования ЦФА

промышленными предприятиями: снижение операционных издержек и рисков, сокращение сроков привлечения финансовых ресурсов. Как вариант дополнительного инвестирования в горнодобывающую и металлургическую промышленность в статье приведен алгоритм использования токенов. Использование цифровых финансовых активов для привлечения финансирования компаниям позволит компаниям получить предварительное авансирование и не обращаться за кредитами в банк. В 2022 г. суммарный об\ем выпуска ЦФА составил около 2 млрд руб, что показывает достаточно серьезную роль ЦФА в настоящее время в сфере привлечения финансирования в компанию, что определяет высокие перспективы развития данного сектора.

Литература

1. Зотов В. М., Абдикеев Н. М. Новые технологии управления финансированием инноваций в промышленности. Финансы: теория и практика; 25(6). DOI: 10.26794/2587-5671-2021-25-6-112-12. Электронный ресурс (2021): https://cyberleninka.ru/article/n/novye-tehnologii-upravleniya-finansirovaniem-innovatsiy-v-promyshlennosti (дата обращения 31.07.2023)
2. Власов М. В. Политика инновационного поведения малых и средних предприятий старопромышленного региона. Экономика региона; 16(4):1335–1347. DOI: 10.17059/ekon.reg.2020-4-22. Электронный ресурс (2020): https://cyberleninka.ru/article/n/politika-innovatsionnogo-povedeniya-malyh-i-srednih-predpriyatiy-staropromyshlennogo-regiona (дата обращения 31.07.2023)
3. Федеральный закон от 31 июля 2020 г. № 259-ФЗ «О цифровых финансовых активах, цифровой валюте и о внесении изменений в отдельные законодательные акты Российской Федерации» (с изменениями и дополнениями). — Справочно-правовая система «ГАРАНТ». — Электронный ресурс: https://base.garant.ru/74451466/ (дата обращения: 31.07.2023)
4. А. Веролайнен, П. Кашицын «Специфические свойства российских цифровых финансовых активов». Официальный сайт РА «Эксперт». 24.03.2023 — Электронный ресурс: https://raexpert.ru/researches/digital_fin_2023/ (дата обращения: 31.07.2023)
5. Бизнес переходит в цифру. Коммерсантъ. Электронный ресурс: 02.03.2021 https:// www.kommersant.ru/doc/4710180 (дата обращения: 31.07.2023).
6. Алексей Погорелый. Токенизация активов горнодобывающей и металлургической промышленности. — 12.06.2020. Электронный ресурс: https://hub.forklog.com/tokenizatsiya-aktivov-gornodobyvayushhej-i-metallurgicheskoj-promyshlennosti/ (дата обращения: 31.07.2023)

Application of digital financial assets in the activities of industrial companies

Aleshina A. V., Demidov A. D.,
Plekhanov Russian Economic University, Lomonosov Moscow State University

The active development of digital financial technologies has become a trend in recent years. 2022 was the period of placement of debut issues of digital financial assets (DFA) in Russia, the total volume of which amounted to about 2 billion rubles. This confirms the success of the implemented strategy to bring together regulators of DFA and traditional financial instruments. In the future, this interaction algorithm is designed to influence the promising product situation on the market and its risk portfolio. Investing in digital financial assets in the current geopolitical and economic environment is extremely risky, first of all, the platform itself does not have strict regulations. This concerns restrictions affecting the compatibility of technologies of various operators. But even taking into account the existing problems, the DFA market has good long-term prospects, especially in the field of automated and controlled transactions based on smart contracts. The article touches on the topic of tokenization of mining enterprises in Russia.

Keywords: digital financial assets, DFA, investments, sources of financing, industrial enterprises.

References

1. Zotov V. M., Abdikeev N. M. New technologies for managing innovation financing in industry. Finance: Theory and Practice; 25(6). DOI: 10.26794/2587-5671-2021-25-6-112-12. Electronic resource (2021): https://cyberleninka.ru/article/n/novye-tehnologii-upravleniya-finansirovaniem-innovatsiy-v-promyshlennosti (date of access 31.07.2023)

2. Vlasov M. V. Policy of innovative behavior of small and medium enterprises of an old industrial region. Economy of the region; 16(4):1335–1347. DOI: 10.17059/ekon.reg.2020-4-22. Electronic resource (2020): https://cyberleninka.ru/article/n/politika-innovatsionnogo-povedeniya-malyh-i-srednih-predpriyatiy-staropromyshlennogo-regiona (date of access 31.07.2023)
3. Federal Law of July 31, 2020 No. 259-FZ «On Digital Financial Assets, Digital Currency and on Amendments to Certain Legislative Acts of the Russian Federation» (with amendments and additions). — Reference and legal system «GARANT». — Electronic resource: https://base.garant.ru/74451466/ (date of access: 31.07.2023)
4. A. Verolaynen, P. Kashitsyn «Specific properties of Russian digital financial assets». Official website of RA «Expert». 24.03.2023 — Electronic resource: https://raexpert.ru/researches/digital_fin_2023/ (date of access: 31.07.2023)
5. Business is going digital. Kommersant. Electronic resource: 02.03.2021 https:// www.kommersant.ru/doc/4710180 (date of access: 31.07.2023).
6. Alexey Pogorely. Tokenization of assets in the mining and metallurgical industry. — 12.06.2020. Electronic resource: https://hub.forklog.com/tokenizatsiya-aktivov-gornodobyvayushhej-i-metallurgicheskoj-promyshlennosti/ (date of access: 31.07.2023)

Децентрализованные финансы и изменения в банковской системе

Дмитриева Людмила Владиславовна
к. э. н.„ первый заместитель Председателя Правительства Ивановской области
доцент ФГБОУ ВО «Ивановский государственный университет»
E-mail: ludmilavd@yandex.ru

Алешина Анна Валентиновна
к. э. н., доцент, экономический факультет МГУ имени М. В. Ломоносова, г. Москва
Российская Федерация
E-mail: annaaaleshina@mail.ru

Развитие такой области финтеха как децентрализованные финансы (DeFi) происходит весьма интенсивно, и данная экосистема бросает вызов традиционному финансово-банковскому сектору. Основой DeFi является блокчейн технологии. Потребителю предлагается широкий выбор финансовых услуг на базе DeFi: займы, инвестиции, кредитование, финансирование торговли и денежные расчеты по сделке. Отличительной чертой всех децентрализованных продуктов является отсутствие посредника в лице банка или иного финансового учреждения. За последние 3–4 года DeFi привлекает внимание многих экспертов, ведь к 2021 г. суммарный объем сделок, проходящих через протоколы DeFi достиг 100 млрд долларов. Пользователи заинтересованы в таких сделках, а транзакции, проводимые в рамках DeFi, позиционируются как максимально прозрачные и надежные. Децентрализованный доступ к финансовым услугам без участия посредников позволяет существенно снизить затраты на проведение финансовых операций, что является дополнительным плюсом внедряемых платформ DeFi.

Ключевые слова: *децентрализованные финансы, блокчейн*, DeFi, цифровые валюты центральных банков (CBDC).

Введение

Традиционные финансовые институты функционируют много столетий и ключевая их особенность — присутствие посредника в виде банков. Суть функционирования всей банковской системы строится на аккумулировании свободных средств вкладчиков на депозитах и сберегательных счетах с последующим предоставлением кредитов лицам, нуждающимся в свободном капитале. Разница в процентах между процентной ставкой по вкладам и процентом, который банк получает по кредиту, образует прибыль финансово-кредитных учреждений. Но риски существуют в любых сферах, и банк может в любой момент обанкротиться, приводя к финансовой несостоятельности своих вкладчиков. В попытках создания более совершенного финансового института законодатели разных стран пытались сузить диапазон оказываемых банковских услуг с одновременным увеличением надежности и безопасности активов.

Под воздействием специфического регулирования финансовых операций появились новые типы инструментов и новые типы финансовых посредников, например, глобальные инвестиционные фонды Люксембурга появились как попытка избежать серьезного банковского регулирования в США и в Великобритании; американские депозитарные расписки (American depository receipts (ADR)) появились как попытка обхода американского законодательства, которое запрещает раз-

мещать на американских биржах акции иностранных компаний. ADR представляют собой американские депозитарные (банковские) расписки, дающие право на определенное количество иностранных акций. Глобальные депозитарные расписки (GDR) появились для размещения иностранных акций на английских и гонконгских биржевых площадках. Новые компании из области финтеха постепенно вытесняют традиционные финансовые институты с рынка, необходимость в наличных денежных средствах и пластиковых картах снижается в связи с более активным распространением DeFi [1] и появлением цифровых валют центральных банков (CBDC). Децентрализованные финансы DeFi стали наиболее ярким примером новых технологий.

DeFi: диапазон возможностей и обзор рисков

Эксперты считают децентрализованные финансы (Decentralized finance — DeFi) новым направлением развития инфраструктуры финансового рынка. Новые технологии, представленные DeFi, обеспечивают уже большинство аналогов классических финансовых операций, которые совершают традиционные финансовые институты, в том числе хранение вкладов под процент, предоставление кредитов, страхование, финансирование торговли, платежи и расчеты между физическими и юридическими лицами, в том числе из разных стран.

При этом перечень сервисов и услуг DeFi постоянно расширяется, появляются новые финансовые продукты, которые конкурируют с классическими банковскими операциями. Потребители финансовых услуг, в первую очередь, заинтересованы в надежности и стабильности финансовых институтов и организаций, поэтому высокорисковая, на первых порах, индустрия DeFi интенсивно перестраивается, предлагая потребителю аналоги классических финансовых услуг, в том числе депозитов и облигаций с фиксированными процентными ставками.

Во многом появление новых альтернативных финансовых инструментов связано с «ростом государственного долга развитых стран, ускорением мировых финансовых кризисов и мировой инфляцией, а также пользовательским поиском источников альтернативной доходности» [2]. Финансовая децентрализация обладает рядом потенциальных возможностей, которые представлены в таблице 1.

Таблица 1. Потенциальные возможности децентрализации финансовых операций

Фактор	Достоинства децентрализованных финансовых операций
Сфера инноваций	Инновационная среда для новых видов финансовых услуг Испытательный полигон для тестирования классических финансовых услуг в цифровой форме
Источник операционной эффективности	Снижение трансакционных издержек Повышение прозрачности финансовых сделок Автоматизация транзакций и рост скорости совершения сделок Снижение риска ликвидности посредника
Возможности для развития экономики	Доступность для широкого круга лиц, в том числе в тех ситуациях, когда доступ к классическим банковским продуктам затруднен Развитие конкуренции среди финансовых организаций из разных стран

DeFi представляет собой потенциал для инноваций. Новый инструмент финтеха выступает испытательным полигоном для традиционного финансового рынка. DeFi также обеспечивают рост операционной эффективности финансового бизнеса [3] в связи с сокращением издержек на проведение платежей и совершение иных финансовых операций, рост автоматизации проведения транзакций, увеличение скорости проведения финансовых операций, повышение прозрачности сделок.

Смарт-контракты позволяют повысить скорость совершения финансовых операций, снижают транзакционные издержки на совершение сделки, так как практически минимизируют необходимость «человеческого» вмешательства в процесс заключения и исполнения финансовой сделки. Использование смарт-контрктов позволяет автоматизировать процессы, что приводит к снижению затрат времени на оформление сделки и отслеживания по сделке платежа, что приводит к снижению риска неисполнения сделки. Так как DeFi позволяют отказаться от использования централизованных доверенных посредников, и благодаря автоматизации приложений, DeFi повышают итоговую скорость проведения сделки от момента ее заключения до окончательного ее исполнения.

Децентрализованные финансы позволяют обеспечить обезличивание финансовых операций. С одной стороны операции благодаря DeFi осуществляются достаточно прозрачно, с другой стороны, можно гарантировать высокую степень анонимности операций при сохранении их относительной прозрачности. Механизм проведения транзакции с использования инструментов DeFi обеспечивает, что: «данные о протоколе и совершаемых внутри него платежах доступны для публичного анализа, но лишь на псевдонимной основе» [1]. Не требуется вмешательство человека в реализацию механизма «smart-контракт — smart-контракт». Это обеспечивает технологическое проведение любых транзакций.

«Запуск операций происходит на базе каналов данных, которые предоставляют внешние узлы (оракулы) или протокол. Очевидный плюс DeFi — пользователи самостоятельно хранят свои активы, что минимизирует риски утраты их посредниками в случае финансового кризиса, операционных сбоев или дефолта» [1]. Дополнительно в силу того, что единая точка отказа или воздействия отсутствует при децентрализации, уровень технологической устойчивости всей системы перед киберрисками потенциально повышается, что повышает привлекательность децентрализованных финансов и приводит к росту операций с криптоактивами (см. рис. 1).

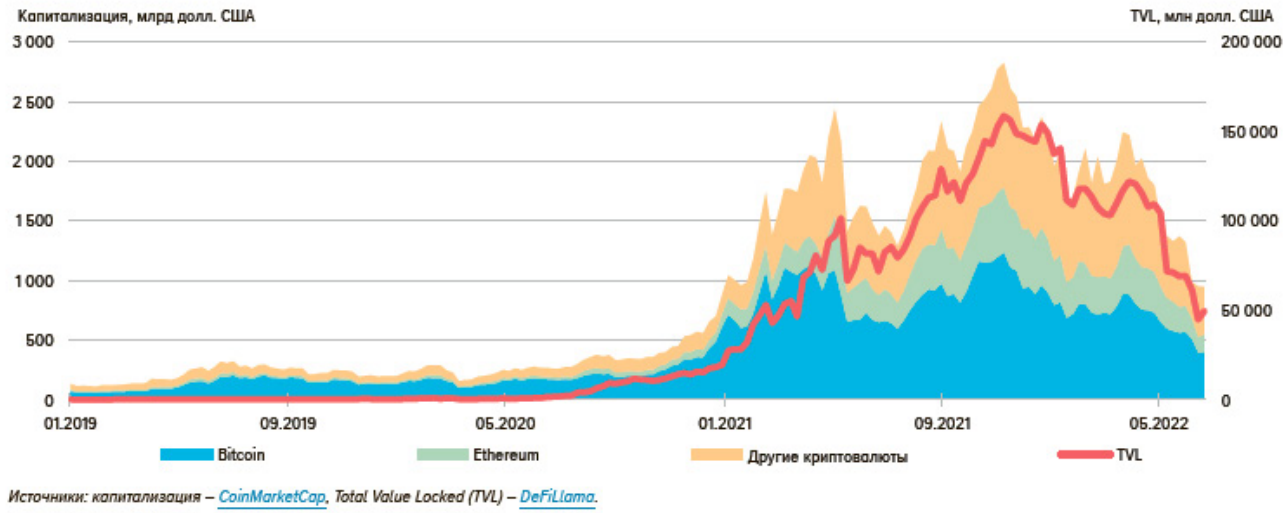

Рис. 1. Динамика оценки капитализации рынка криптоактивов и общей стоимости в DeFi (ETH) [3, с. 7].

Третьим пунктом в линейке потенциальных возможностей DeFi стоит доступность для общества, проявляющаяся в получении доступа к финансовым услугам из любого уголка мира, независимо от статуса (компания или частное лицо) и географического местоположения. Однако, стоит заметить, что пользователь должен обладать не только точкой входа в интернет, он априори обязан обладать определенными компетенциями в сфере DeFi. Иначе говоря, он должен иметь достаточный уровень финансовой грамотности и ментальную потребность (готовность) в использовании данного рода инструментов.

Несмотря на позиционирование DeFi как способа получения финансовых услуг всеми слоями населения, включая не охваченных банковским обслуживанием, на практике данные финансовые технологии получили распространение пока только в странах с высоким уровнем доходов граждан. Для пользователей финансовых услуг из этих стран проблема финансовой доступности не стоит столь остро, как для стран с более низким охватом финансовых институтов.

Благодаря созданию возможностей для использования самых разных продуктов обеспечивается внешняя конкуренция децентрализованных и традиционных финансовых услуг. Например, существует возможность использования крупных децентрализованных бирж (decentralized exchange — DEX) для конвертации средств в стейблкоины, обеспеченные долларами и предназначенными для трансграничных денежных транзакций или финансирования протоколов DeFi-кредитования.

Децентрализованные биржи (decentralized exchange — DEX) отличаются от традиционных площадок с организованными торгами и централизованных бирж (centralized exchange — CEX) тем, что хранение активов, клиринг и сопоставление заявок осуществляются по алгоритму, заложенному в коде smart-контракта. Происходящие в режиме реального времени и без привлечения посредников операции гарантируют пользователям DEX частичное снижение издержек и рисков. Тот факт, что такой тип финансовых услуг востребован, подтверждается неуклонным ростом объемов торгов на DEX год от года (см. рис. 2).

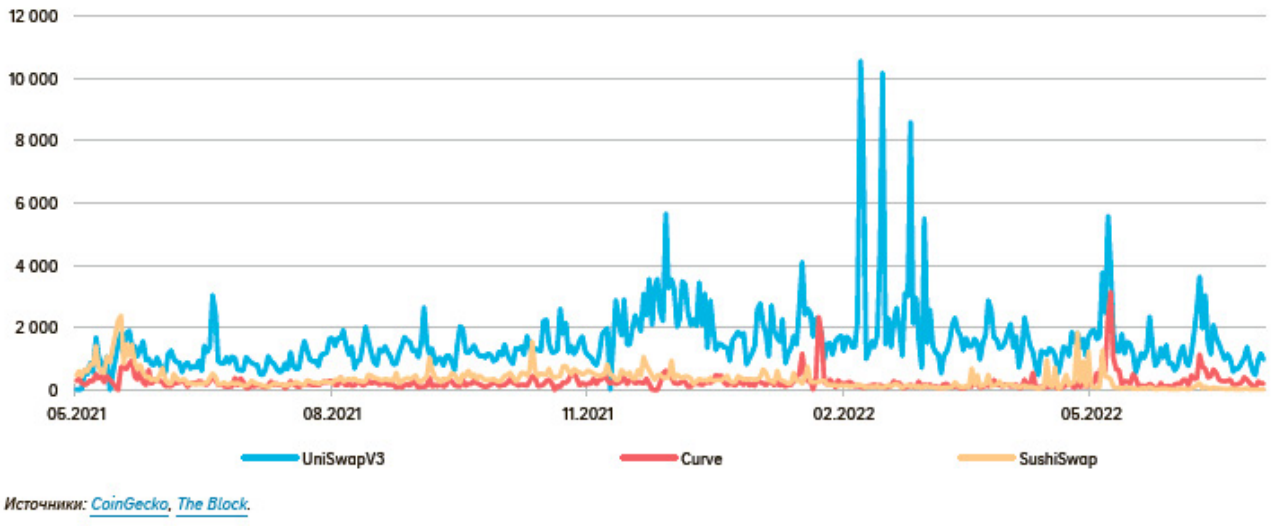

Рис. **2.** Объемы торговли на крупнейших децентрализованных биржах (DEX) [3, с. 24].

Усиление конкуренции между рынком традиционных и децентрализованных финансовых услуг обусловлено постоянным наращиванием сетевых эффектов и, как следствие, перетоком клиентов из первого во второй сегмент. Помимо внешней конкуренции, в сфере децентрализованных финансов присутствует внутрисетевая конкуренция. Она усиливается

из-за децентрализованного формата и большого рассредоточения поставщиков услуг. Последнее обстоятельство и обеспечивает разнообразие при одновременном снижении уровня концентрации данных поставщиков.

Рассматривая риски, связанные с DeFi, можно отметить несколько достаточно важных аспектов:

- отсутствие классических механизмов надзора и регулирования, что увеличивает риск мошеннических операций и создания специальных мошеннических компаний;
- недостаточно проработанные на законодательном уровне требования к системам управления рисками, что увеличивает риск банкротства таких организаций;
- отсутствие обеспечения необходимого риск-контроля со стороны текущих функций саморегулирования;
- при росте автоматизированных продуктов DeFi может значительно усилиться полицикличность;
- за счет потенциальной корреляции между интенсивно растущими криптоактивами и децентрализованными финансами может повыситься уровень левериджа;
- неконтролируемый переток участников из сегмента традиционных финансовых услуг в нерегулируемый DeFi сегмент может спровоцировать «эффект домино» для многих крупных международных инвесторов, когда банкротство одного крупного финансового института может повлечь за собой цепочку банкротств следующих финансовых институтов [4].

Если потребители потеряют часть активов в результате негативного опыта в проектах с децентрализованными финансами, уровень их доверия к цифровым финансовым инструментам может заметно снизиться. Поэтому полный переход на альтернативную и нерегулируемую платежную инфраструктуру в среде DeFi сегодняшний день представляется крайне рискованным шагом.

Будущее DeFi, или какие изменения в банковской сфере скоро станут реальностью

Несмотря на имеющиеся риски, децентрализованные финансы способны произвести в банковском секторе настоящую революцию. В частности, речь идет о трансформации форм и направлений использования привычных человечеству финансовых услуг. Большинство экспертов развитие DeFi в перспективе связывают со следующими тенденциями:

- дальнейшее расширение сектора DeFi. Общая стоимость платформ с децентрализованными финансами составляет уже более 100 млрд долларов и за счет интеграции с традиционным финансовым сектором рост будет продолжаться [5];
- ожидание в ближайшем будущем совместимости платформ DeFi между собой, а также с общемировыми банковскими системами;
- работа разработчиков над улучшением пользовательского интерфейса и упрощения финансовых инструментов сделают DeFi доступными более широкому кругу пользователей цифровыми финансовыми услугами;
- необходимость более тщательного законодательного регулирования, скорее всего, привлечет внимание регулирующих органов и мировым юрисдикциям придется вводить дополнительные правила и разумные ограничения и устанавливать зону ответственности;
- на рынке скоро появятся новые приложения DeFi, которые будут представлять собой конкуренцию классическим финансовым операциям через банки и другие финансовые посредники как в сфере хранения денег, в сфере переводов денег за рубеж и в сфере децентрализованного страхования. В ближайшей перспективе на платформах DeFi будут внедрены инновационные технологии, которые лягут в основу приложений и вариантов использования (NFT);

- ожидается интеграция платформ DeFi с другими передовыми технологиями вроде искусственного интеллекта и интернета вещей (IoT).

Если рассматривать вопрос влияния децентрализованных финансов на банковский сектор, то здесь стоит отметить несколько ключевых направлений:

1) устранение посредников. Благодаря снижению транзакционных издержек на платформах DeFi растет не только эффективность совершаемых операций, но и пропадает потребность в банках и иных финансово-кредитных учреждениях;
2) малообеспеченные и небанковские пользователи получают доступ к таким финансовым услугам, которые им в традиционном финансовом сегменте недоступны;
3) прозрачность переводов повышает доверие пользователей, однако, случаи банкротств финансовых компаний, оказывающих услуги в секторе DeFi, уже происходили, поэтому утверждать, что прозрачность равна безопасности тоже нельзя;
4) за счет высокой степени программируемости платформы DeFi могут перенастраиваться, что повышает адаптивность данных инструментов в сфере финансовых услуг;
5) глобальный охват открывает доступ к децентрализованным финансам людям из разных стран, единственным условием получения услуг становится наличие точки подключения к интернету.

Расширение экосистемы DeFi гарантирует высокий потенциал и способность преобразования всего банковского сектора услуг. Несмотря на то, что система DeFi находится в начале развития, финансовые институты и частные инвесторы уже инвестировали в нее огромные средства, что говорит о высоких перспективах роста.

Выводы:

Децентрализованные финансы способны преобразовать финансовый сектор, который многие столетия обеспечивал пользователей разнообразными услугами и продуктами. DeFi обладают глобальным охватом, за счет отсутствия посредников обеспечивают прозрачность транзакций. Для масштабируемости цифрового сегмента финансов необходимо сохранение пользовательского доверия и интереса, обеспечение функциональной совместимости платформ и введение разумного алгоритма правового регулирования. Стоит отметить, что банковская сфера не перестанет существовать в один момент, для нее появление альтернативной экосистемы должно стать вызовом, стимулом к развитию и толчком к поиску вариантов интеграции с инновационными технологиями.

Литература

1. Алешина А. В.,Булгаков А. Л. Децентрализованные финансы (DeFi): риски, перспективы и регулирование. Финансовые рынки и банки. № 12, 2022. Электронный ресурс: https://cyberleninka.ru/article/n/detsentralizovannye-finansy-defi-riski-perspektivy-i-regulirovanie
2. De Filippi P., Loveluck B. The invisible politics of bitcoin: governance crisis of a decentralized infrastructure // Internet Policy Review. — 2016. — Т. 5. — № 4. Электронный ресурс: https://www.researchgate.net/publication/309574531_The_invisible_politics_of_Bitcoin_governance_crisis_of_a_decentralised_infrastructure
3. Децентрализованные финансы. Официальный сайт Банка России: www.cbr.ru, 2022. Электронный ресурс: https://cbr.ru/Content/Document/File/141992/report_07112022.pdf/
4. Decentralized Finance (DeFi) and Challenges to Traditional Financial System. 08 Apr, 2023. Электронный ресурс: https://www.geeksforgeeks.org/decentralized-finance-defi-and-challenges-to-traditional-financial-system/

Decentralized Finance and Changes in the Banking System

Dmitrieva L. V., Aleshina A. V.
Ivanovo State University, Lomonosov Moscow State University

The development of such a fintech area as decentralized finance (DeFi) is very intensive, and this ecosystem challenges the traditional financial and banking sector. DeFi is based on blockchain technology. The consumer is offered a wide range of financial services based on DeFi: loans, investments, lending, trade financing and cash settlements for transactions. A distinctive feature of all decentralized products is the absence of an intermediary in the person of a bank or other financial institution. Over the past 3–4 years, DeFi has attracted the attention of many experts, because by 2021 the total volume of transactions through DeFi protocols reached $100 billion. Users are interested in such transactions, and transactions carried out within DeFi are positioned as the most transparent and reliable. Decentralized access to financial services without the participation of intermediaries can significantly reduce the cost of financial transactions, which is an additional advantage of the implemented DeFi platforms.

Keywords: decentralized finance, blockchain, DeFi, central bank digital currencies (CBDC).

References

1. Aleshina A. V., Bulgakov A. L. Decentralized finance (DeFi): risks, prospects and regulation. Financial markets and banks. No. 12, 2022. Electronic resource: https://cyberleninka.ru/article/n/detsentralizovannye-finansy-defi-riski-perspektivy-i-regulirovanie
2. De Filippi P., Loveluck B. The invisible politics of bitcoin: governance crisis of a decentralized infrastructure // Internet Policy Review. — 2016. — Vol. 5. — No. 4. Electronic resource: https://www.researchgate.net/publication/309574531_The_invisible_politics_of_Bitcoin_governance_crisis_of_a_decentralised_infrastructure
3. Decentralized finance. Official website of the Bank of Russia: www.cbr.ru, 2022. Electronic resource: https://cbr.ru/Content/Document/File/141992/report_07112022.pdf/
4. Decentralized Finance (DeFi) and Challenges to Traditional Financial System. 08 Apr, 2023. Electronic resource: https://www.geeksforgeeks.org/decentralized-finance-defi-and-challenges-to-traditional-financial-system/
5. Results of the study of market opinion on the development of financial technologies for 2021–2023, FinTech Association (AFT) 08/31/2021 // Accenture. Electronic resource //URL: https://www.fintechru.org/analytics/rezultaty-issledovaniya-mneniya-rynka-po-voprosam-razvitiya-finansovykh-tekhnologiy-na-2021–2023-gg-/

Методы анализа данных и подготовка к моделированию в кредитной сфере

Демидов Алексей Дмитриевич
магистр, факультет Высшая школа кибертехнологий, математики и статистики
РЭУ имени Г. В. Плеханова
E-mail: demidov.ad@bk.ru

Булгаков Андрей Леонидович
к. э. н., профессор МИСАО
доцент РЭУ имени Г. В. Плеханова
E-mail: bulgakoval@my.msu.ru

В статье рассматривается процесс анализа данных и подготовки к моделированию для прогнозирования кредитоспособности клиентов банковских учреждений. Представлен обзор современных подходов к обработке данных, включая выявление и устранение аномалий, балансировку классов и визуализацию ключевых характеристик. Проведен подробный анализ набора данных, включая исследование категориальных и числовых признаков, а также их влияние на вероятность получения кредита. Описаны основные этапы подготовки данных для машинного обучения, такие как очистка, трансформация и создание новых признаков. В работе подчеркивается важность качественной предобработки данных для повышения точности прогнозов и эффективности построения моделей. Результаты исследования составляют основу для последующего применения алгоритмов машинного обучения и анализа их производительности в задачах кредитования.

Ключевые слова: скоринговые системы кредитования, анализ данных, кредитование.

Введение

Современные финансовые институты сталкиваются с необходимостью принятия быстрых и обоснованных решений в условиях жесткой конкуренции и нестабильности рынка. Одним из ключевых аспектов их деятельности является эффективное управление процессом кредитования, что позволяет минимизировать риски и максимизировать доходность. Согласно последним исследованиям Мишиной М. Ю., Зверева А. В. (2021) [4] и Далингера К. Е., Айграшевой А. А. (2021) [1], ипотечное кредитование продолжает играть ключевую роль в развитии банковского сектора. В этой связи особую актуальность приобретает использование методов анализа данных и машинного обучения для прогнозирования поведения клиентов.

Особое значение в процессе кредитования имеют индивидуальные характеристики клиентов, такие как их финансовое положение, поведенческие факторы и предыдущий опыт взаимодействия с банком. Например, длительность последнего контакта, количество предыдущих запросов и остаток на счете могут служить важными индикаторами готовности клиента принять предложение.

Важно понимать, что эффективный анализ этих факторов не только повышает точность прогнозов, но и позволяет финансовым организациям разрабатывать более целевые маркетинговые стратегии, оптимизировать взаимодействие с клиентами и сокращать издержки.

Таким образом, использование методов машинного обучения для анализа данных о кредитовании открывает широкие возможности для финансовых организаций, что подчеркивает важность дальнейших исследований в области кредитования Далингера К. Е. и Айграшевой А. А. (2021) [1], а также Мыцык И. В. и Кузнецовой Е. В. (2023) [2]. Эти методы позволяют автоматизировать процесс принятия решений, повысить его обоснованность и ускорить реализацию бизнес-задач. Изучение факторов, влияющих на принятие кредитного предложения, и построение моделей для прогнозирования такого поведения — важная задача, требующая детального анализа данных и оценки производительности различных алгоритмов.

Анализ набора данных

Для начала импортируем библиотеки.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

После этого импортируем набор данных.

```
df = pd.read_csv(«../content/bank2.csv»)
df
```

```
age job marital education default
balance housing loan \
0 59 admin. married secondary no 2343
yes no
1 56 admin. married secondary no 45 no
no
2 41 technician married secondary no
1270 yes no
3 55 services married secondary no 2476
yes no
4 54 admin. married tertiary no 184 no
no
… … … … … … … … …
11157 33 blue-collar single primary no 1
yes no
11158 39 services married secondary no
733 no no
11159 32 technician single secondary no
29 no no
11160 43 technician married secondary no
0 no yes
11161 34 technician married secondary no
0 no no

contact day month duration campaign
pdays previous poutcome \
0 unknown 5 may 1042 1–1 0 unknown
1 unknown 5 may 1467 1–1 0 unknown
2 unknown 5 may 1389 1–1 0 unknown
3 unknown 5 may 579 1–1 0 unknown
4 unknown 5 may 673 2–1 0 unknown
… … … … … … … … …
11157 cellular 20 apr 257 1–1 0 unknown
11158 unknown 16 jun 83 4–1 0 unknown
11159 cellular 19 aug 156 2–1 0 unknown
11160 cellular 8 may 9 2 172 5 failure
11161 cellular 9 jul 628 1–1 0 unknown
deposit
0 yes
1 yes
2 yes
3 yes
4 yes
… …
11157 no
11158 no
11159 no
11160 no
11161 no
[11162 rows x 17 columns]
```

Проверяем значения набора данных на Null.

```
df.isnull().sum()
age 0
job 0
marital 0
education 0
default 0
balance 0
housing 0
loan 0
contact 0
day 0
month 0
duration 0
campaign 0
pdays 0
previous 0
poutcome 0
deposit 0
dtype: int64
```

В результате не получили нулевых значений, которые могли бы внести искажения в расчеты.

Проверяем значения набора данных на дублирующиеся записи.

```
df.duplicated().sum()
0
```

Нет дублирующийся значений, поэтому не нужно их исключать из набора данных.

Анализ типов данных

Получаем типы данных нашего набора данных.

```
df.dtypes
age int64
job object
marital object
education object
default object
balance int64
housing object
loan object
contact object
day int64
month object
duration int64
campaign int64
pdays int64
previous int64
poutcome object
deposit object
dtype: object
df.describe()
age balance day duration campaign \
count 11162.000000 11162.000000 11162.00
0000 11162.000000 11162.000000
mean 41.231948 1528.538524
15.658036 371.993818 2.508421
std 11.913369 3225.413326
8.420740 347.128386 2.722077
min 18.000000–6847.000000 1.000000
2.000000 1.000000
25% 32.000000 122.000000
8.000000 138.000000 1.000000
50% 39.000000 550.000000
15.000000 255.000000 2.000000
75% 49.000000 1708.000000
22.000000 496.000000 3.000000
max 95.000000 81204.000000
31.000000 3881.000000 63.000000

pdays previous
count 11162.000000 11162.000000
mean 51.330407 0.832557
std 108.758282 2.292007
min–1.000000 0.000000
25%–1.000000 0.000000
50%–1.000000 0.000000
75% 20.750000 1.000000
max 854.000000 58.000000
```

Баланс данных

Проверяем сбалансированны ли данные. Для этого необходимо разделить их общее количество на то, дали ли кредит человеку или нет.

```
df[‘deposit’].value_counts()/df.shape[0]
no 0.52616
yes 0.47384
Name: deposit, dtype: float64
```

```
plt.figure(figsize=(12,5))
plt.subplot(1,2,1)
sns.countplot(x='deposit', data=df)
plt.title(«Сколько человек получили кре-
дит»)
plt.subplot(1,2,2)
labels =df['deposit'].value_counts(sort
= True).index
sizes = df['deposit'].value_counts(sort
= True)
plt.pie(sizes, labels=labels,
autopct='%1.1f%%', shadow=True,
startangle=270,)
plt.title('Процентное соотношение', size
= 12)
plt.show()
```

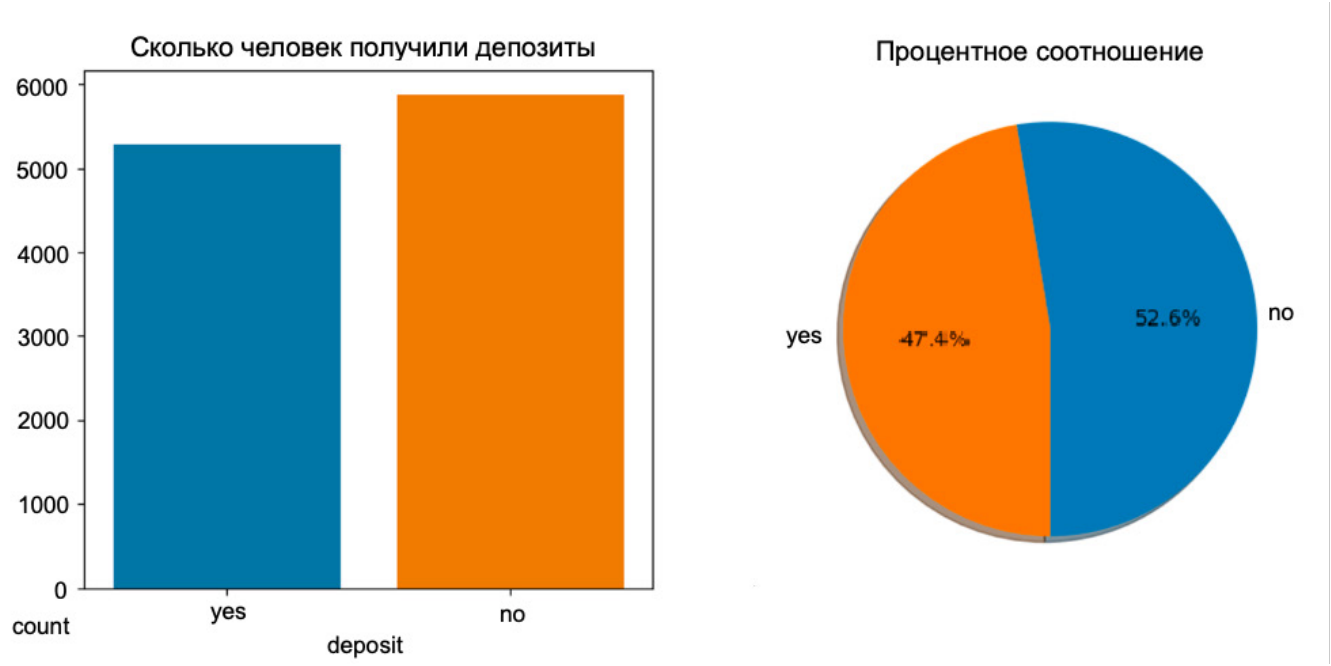

Рис. 1. Процентное соотношение лиц, получивших кредит (составлено авторами)

Как мы можем увидеть (см. рис. 1), данные сбалансированные. В противном случае их можно было бы сбалансировать путем случайной передискретизации.

Проведем визуализацию данных и их приведение к нужному формату.

Рассмотрим данные с типом Object (см. рис. 2 и 3).

```
df_cat = df.select_
dtypes(include='object').columns.
drop([«deposit»,»job»])
df_cat
Index(['marital', 'education',
'default', 'housing', 'loan', 'contact',
'month', 'poutcome'],
dtype='object')
plt.figure(figsize=(13,15))
for i, cat_fea in enumerate(df_cat):
plt.subplot(4,2, i+1)
sns.countplot(x=cat_fea, hue='deposit',
data=df, edgecolor=»black»)
plt.title(«Статистика по {}».format(cat_
fea))
plt.tight_layout()
plt.show()
plt.figure(figsize=[14,5])
sns.countplot(x='job', hue='deposit',
edgecolor=»black», data=df)
plt.title(«Статистика по занимаемым ва-
кансия»)
plt.show()
```

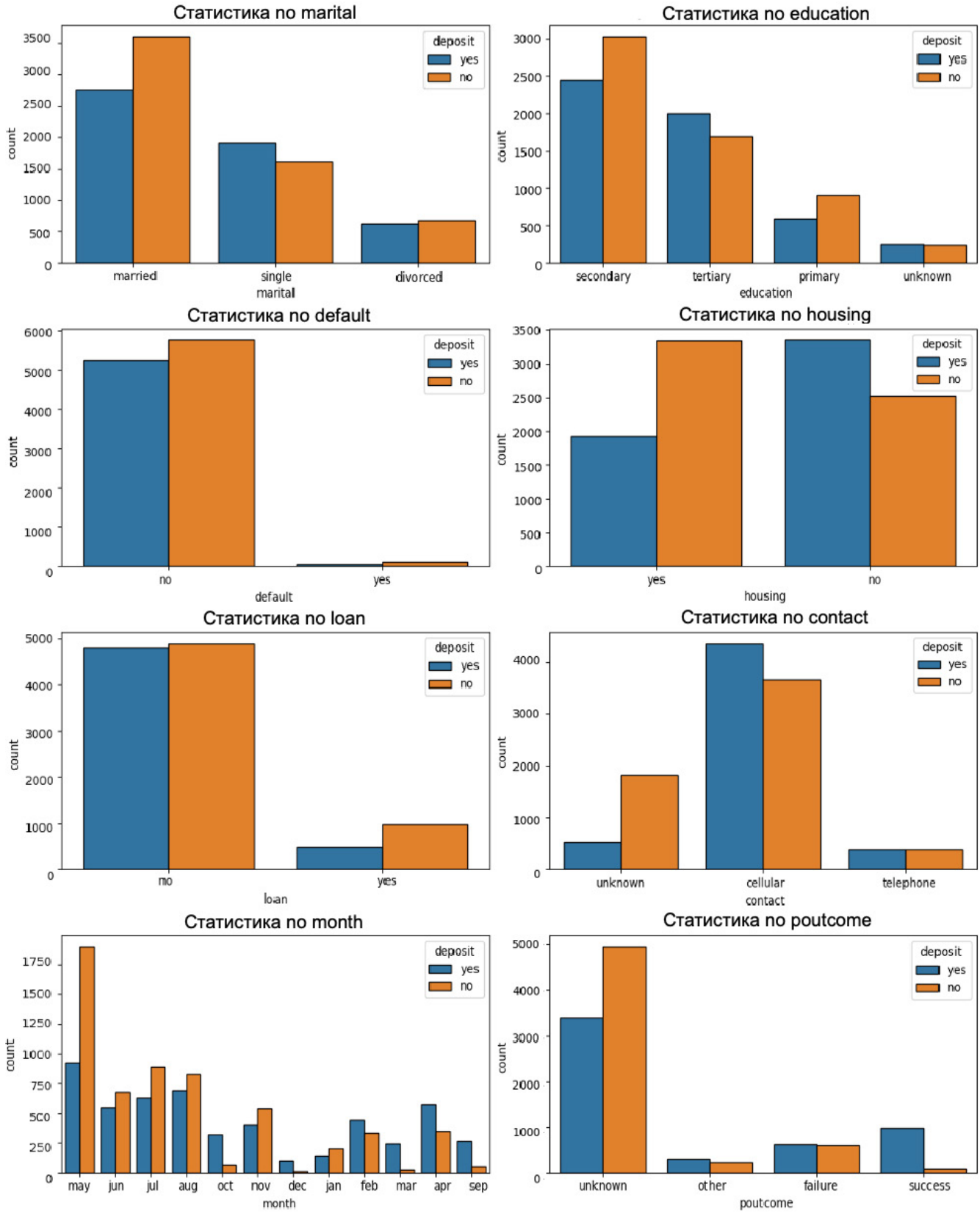

Рис. 2. Распределение категориальных признаков по параметру «Получение кредита» (составлено авторами)

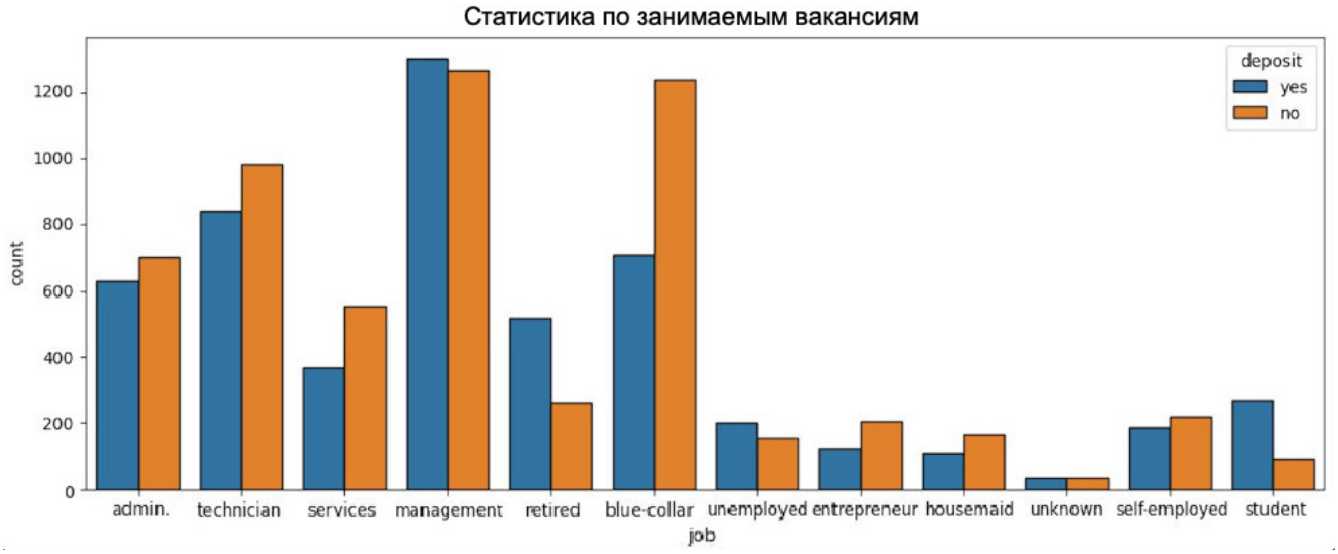

Рис. 3. **Статистика по распределению профессий среди получателей кредита (составлено авторами)**

Выводы

1) Одинокий человек с большей вероятностью возьмет кредит, нежели чем человек в браке.

2) Люди без ипотеки с большей вероятностью возьмут кредит

3) По сотовому телефону человек с большей вероятностью возьмет кредит, нежели чем по стационарному

4) В мае было сделано больше всего звонков, а именно 2500. В декабре меньше всего, а именно 200

5) Студенты и пенсионеры чаще брали срочные кредиты.

Рассмотрим данные с типом Num.

В этом нам поможет точечная диаграмма (см. рис. 4).

```
df_num = df[[‘age’, ‘balance’, ‘day’,
‘duration’, ‘campaign’, ‘pdays’,
‘previous’,’deposit’]]
col = [‘age’, ‘balance’, ‘day’,
‘duration’, ‘campaign’, ‘pdays’,
‘previous’,’deposit’]
plt.figure(figsize=(15,18))
for i, v in enumerate(col):
print(i, v)
plt.subplot(4,2, i+1)
sns.scatterplot(x=v, y=’deposit’,
data=df_num, color=’blue’)
plt.title(«Диаграмма рассенивания по
{} к кредиту».format(v), size=20,
color=»red»)
plt.xlabel(«{}».format(v), size=15)
plt.ylabel(«Deposit», size=15)
plt.tight_layout()
plt.show()
```

```
0 age
1 balance
2 day
3 duration
4 campaign
5 pdays
6 previous
7 deposit
```

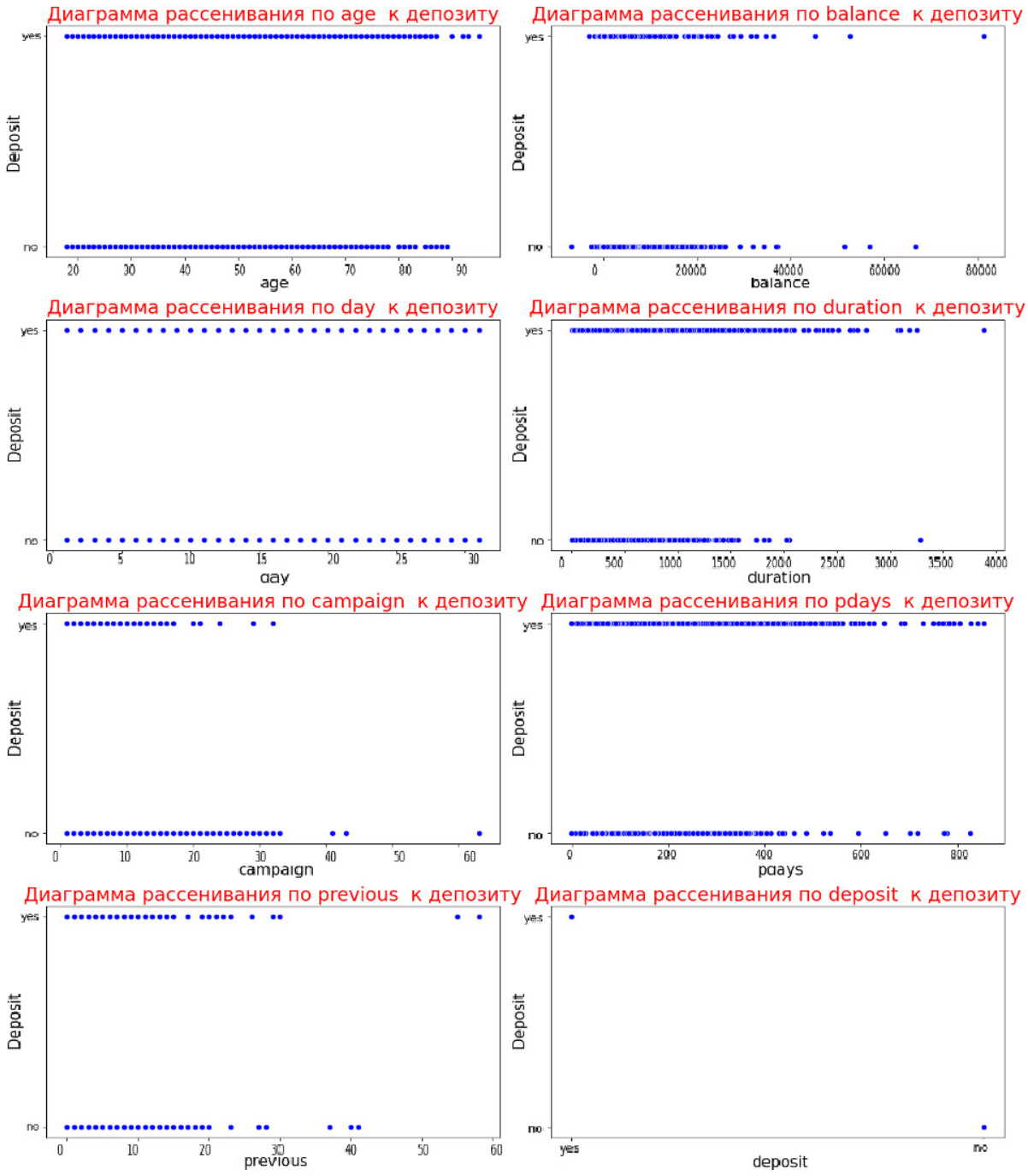

Рис. 4 **Диаграммы рассеяния ключевых числовых переменных, влияющих на вероятность получения кредита**

Диаграмма «Box plot»

Если набор данных небольшой, мы можем обнаружить отклонение, просто взглянув на него.

Но если набор данных большой, то нам нужно использовать визуаизацию и математические методы.

Ниже приводятся некоторые методы обнаружения отклонений (см. рис. 5).

```
df_num = df[['age', 'balance', 'day',
'duration', 'campaign', 'pdays',
'previous']]
col = ['age', 'balance', 'day',
'duration', 'campaign', 'pdays',
'previous']
plt.figure(figsize=(15,18))
for i, v in enumerate(col):
print(i, v)
```

```
plt.subplot(4,2, i+1)
sns.boxplot(x=v, data=df_num,
color='green')
plt.title(«Boxplot для {}».format(v),
size=20, color=»red»)
plt.xlabel(«{}».format(v), size=15)
plt.tight_layout()
plt.show()
```

```
0 age
1 balance
2 day
3 duration
4 campaign
5 pdays
6 previous
```

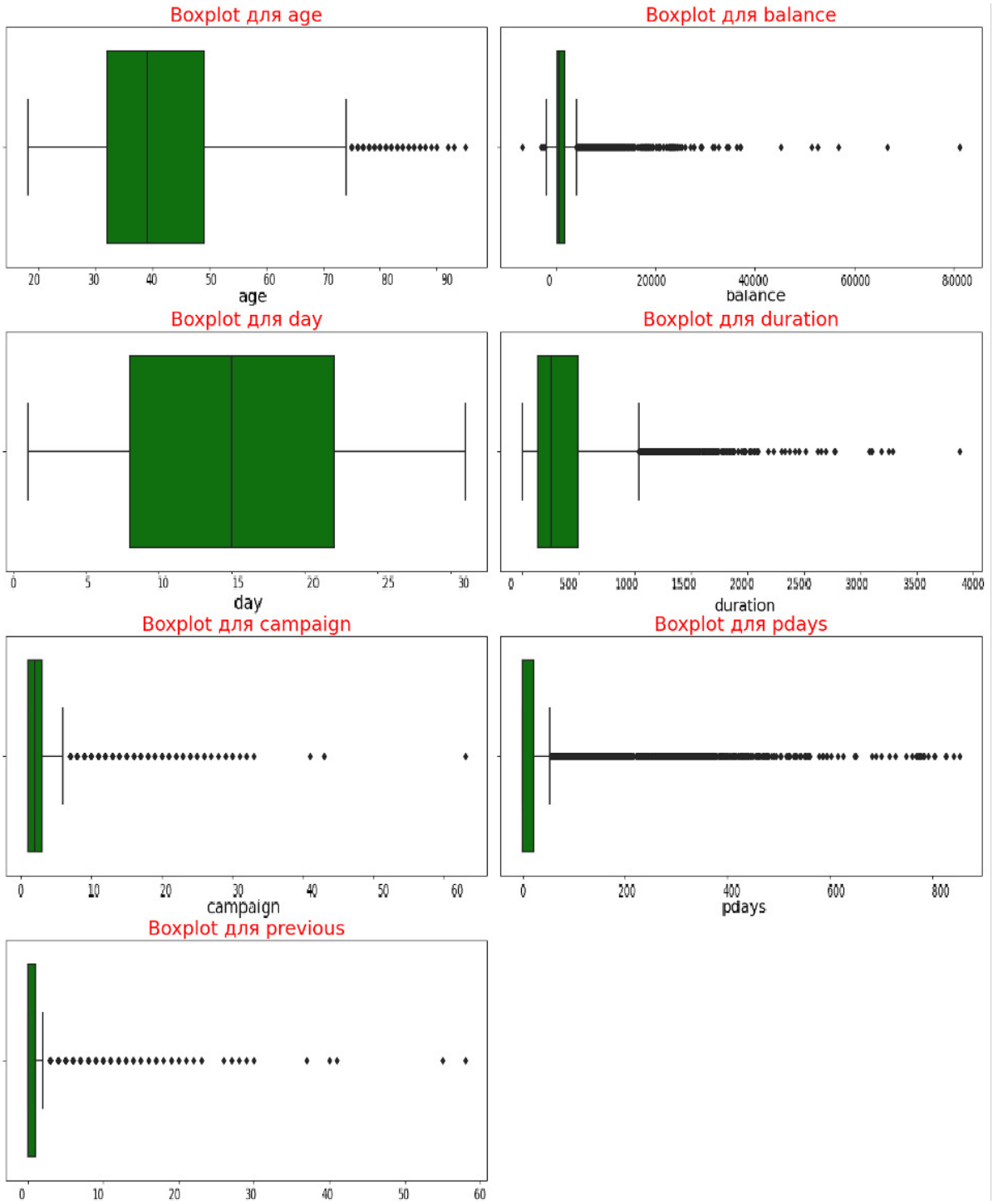

Рис. 5. Boxplot-диаграммы для визуализации распределения значений числовых переменных

Выводы

1) Чем выше продолжительность разговора, тем вероятнее, что человек возьмет кредит.

2) Чем меньше звонков поступило к клиенту, тем ниже шансы того, что он возьмет кредит.

3) Люди, взявшие кредит, имеют больше денег на счету.

Обработка данных

В процессе обработки данных особое внимание было уделено очистке переменных с аномальными значениями и выбросами, такими как balance, duration, campaign и previous. Это связано с тем, что наличие экстремальных значений может значительно искажать работу алгоритмов машинного обучения и снижать качество предсказаний.

Баланс

Отрицательный баланс в контексте кредитования вызывает сомнения с точки зрения интерпретации. Такие данные могут быть результатом ошибок сбора информации или выбросов, которые не соответствуют логике задачи. Аномально высокие значения смещают средние показатели переменной, что влияет на масштабирование данных и ухудшает работу моделей, особенно чувствительных к расстояниям (например, KNN и SVM). Наиболее значимые параметры, такие как продолжительность последнего контакта и остаток на счете, подтверждают результаты исследований Мыцык И. В. и Кузнецовой Е. В. (2023) [2].

```
len(df[df['balance']<0])/len(df)
0.061637699337036375
df[(df['balance']>40000)|(df['balan
ce']<0)]
age job marital education default
balance housing loan \
17 49 services married secondary no-8
yes no
23 43 blue-collar married primary no-192
yes no
30 32 blue-collar married secondary
yes-1 yes no
42 45 entrepreneur divorced tertiary
no-395 yes no
59 57 technician married tertiary no-1
no no
… … … … … … … … …
11119 39 management married tertiary
no-974 no yes
11120 50 management married tertiary
no-516 yes no
11132 32 blue-collar married secondary
no-325 yes yes
11145 60 retired divorced tertiary
no-134 no no
11156 34 blue-collar single secondary
no-72 yes no

contact day month duration campaign
pdays previous poutcome \
17 unknown 8 may 1119 1-1 0 unknown
23 unknown 8 may 1120 2-1 0 unknown
30 unknown 9 may 653 1-1 0 unknown
42 unknown 13 may 470 1-1 0 unknown
59 unknown 14 may 850 2-1 0 unknown
… … … … … … … … …
11119 cellular 13 aug 130 4-1 0 unknown
11120 unknown 15 may 226 2-1 0 unknown
11132 unknown 21 may 171 1-1 0 unknown
11145 cellular 12 may 243 1 271 4
failure
11156 cellular 7 jul 273 5-1 0 unknown

deposit
17 yes
23 yes
30 yes
42 yes
59 yes
… …
11119 no
11120 no
11132 no
11145 no
11156 no
[696 rows x 17 columns]
```

Как оказалось, у 6% людей отрицательный баланс. Нужно удалить данные записи из набора данных.

Также есть много записей с очень высоким балансом. Их необходимо удалить, чтобы избавится от выбросов.

```
df.drop(df[(df['balance']>40000)|(df['balance']<0)].index, inplace=True, axis=0)
```

duration (продолжительность последнего контакта в секундах)

Переменная duration показывает длительность разговора в секундах и демонстрирует сильное влияние на целевую переменную deposit. Однако были обнаружены аномально высокие значения, превышающие 3000 секунд (пример: запись с длительностью 3881 секунду, или около 65 минут).

```
df[df['duration']>3000]
```

```
age job marital education default
balance housing loan \
153 44 services divorced secondary no 51
yes yes
271 59 management married secondary no
1321 no no
1351 47 blue-collar married secondary no
238 yes yes
4364 53 admin. married secondary no 849
yes no
7198 30 admin. married secondary no 1310
no no

contact day month duration campaign
pdays previous poutcome \
153 unknown 27 may 3094 2–1 0 unknown
271 unknown 9 jun 3881 3–1 0 unknown
1351 cellular 13 mar 3076 1–1 0 unknown
4364 cellular 6 feb 3102 3–1 0 unknown
7198 telephone 27 oct 3284 1–1 0 unknown

deposit
153 yes
271 yes
1351 yes
4364 yes
7198 no
```

Отбросим записи с duration более 3000 т. к. они действуют, как основные выбросы.

```
df.drop(df[df['duration']>3000].index, inplace=True, axis=0)
```

campaign (количество контактов в рамках этой кампании)

Переменная campaign отражает количество контактов с клиентом в текущей маркетинговой кампании. Были выявлены значения, превышающие 40 контактов (например, запись с 63 контактами).

```
df[df['campaign']>40]
```

```
age job marital education default
balance housing loan \
6927 51 blue-collar married unknown no
41 yes no
7139 42 blue-collar married primary no
170 yes no
7240 33 blue-collar married secondary no
0 yes yes
7635 45 management married unknown no
9051 yes no

contact day month duration campaign
pdays previous poutcome \
6927 telephone 9 jul 16 43–1 0 unknown
7139 unknown 19 may 51 41–1 0 unknown
7240 cellular 31 jul 16 43–1 0 unknown
7635 unknown 19 may 124 63–1 0 unknown

deposit
6927 no
7139 no
7240 no
7635 no
```

Удалим основные выбросы по рекламным кампаниям.

```
df.drop(df[df['campaign']>30].index, axis=0, inplace=True)
```

pdays (Количество дней с последнего контакта)

```
df[df['pdays']==-1]
```

```
age job marital education default
balance housing loan \
```

```
0 59 admin. married secondary no 2343
yes no
1 56 admin. married secondary no 45 no
no
2 41 technician married secondary no
1270 yes no
3 55 services married secondary no 2476
yes no
4 54 admin. married tertiary no 184 no
no
… … … … … … … … …
11154 52 technician married tertiary no
523 yes yes
11157 33 blue-collar single primary no 1
yes no
11158 39 services married secondary no
733 no no
11159 32 technician single secondary no
29 no no
11161 34 technician married secondary no
0 no no

contact day month duration campaign
pdays previous poutcome \
0 unknown 5 may 1042 1–1 0 unknown
1 unknown 5 may 1467 1–1 0 unknown
2 unknown 5 may 1389 1–1 0 unknown
3 unknown 5 may 579 1–1 0 unknown
4 unknown 5 may 673 2–1 0 unknown
… … … … … … … … …
11154 cellular 8 jul 113 1–1 0 unknown
11157 cellular 20 apr 257 1–1 0 unknown
11158 unknown 16 jun 83 4–1 0 unknown
11159 cellular 19 aug 156 2–1 0 unknown
11161 cellular 9 jul 628 1–1 0 unknown

deposit
0 yes
1 yes
2 yes
3 yes
4 yes
… …
11154 no
11157 no
11158 no
11159 no
11161 no
[7705 rows x 17 columns]
df[df['poutcome']=='unknown']

age job marital education default
balance housing loan \
0 59 admin. married secondary no 2343
yes no
1 56 admin. married secondary no 45 no
no
2 41 technician married secondary no
1270 yes no
3 55 services married secondary no 2476
yes no
4 54 admin. married tertiary no 184 no
no
… … … … … … … … …
11154 52 technician married tertiary no
523 yes yes
11157 33 blue-collar single primary no 1
yes no
11158 39 services married secondary no
733 no no
11159 32 technician single secondary no
29 no no
11161 34 technician married secondary no
0 no no

contact day month duration campaign
pdays previous poutcome \
0 unknown 5 may 1042 1–1 0 unknown
1 unknown 5 may 1467 1–1 0 unknown
2 unknown 5 may 1389 1–1 0 unknown
3 unknown 5 may 579 1–1 0 unknown
4 unknown 5 may 673 2–1 0 unknown
… … … … … … … … …
11154 cellular 8 jul 113 1–1 0 unknown
11157 cellular 20 apr 257 1–1 0 unknown
11158 unknown 16 jun 83 4–1 0 unknown
11159 cellular 19 aug 156 2–1 0 unknown
11161 cellular 9 jul 628 1–1 0 unknown

deposit
0 yes
1 yes
2 yes
3 yes
4 yes
… …
```

```
11154 no
11157 no
11158 no
11159 no
11161 no
[7707 rows x 17 columns]
df[df['previous']==0]

age job marital education default
balance housing loan \
0 59 admin. married secondary no 2343
yes no
1 56 admin. married secondary no 45 no
no
2 41 technician married secondary no
1270 yes no
3 55 services married secondary no 2476
yes no
4 54 admin. married tertiary no 184 no
no
… … … … … … … … …
11154 52 technician married tertiary no
523 yes yes
11157 33 blue-collar single primary no 1
yes no
11158 39 services married secondary no
733 no no
11159 32 technician single secondary no
29 no no
11161 34 technician married secondary no
0 no no

contact day month duration campaign
pdays previous poutcome \
0 unknown 5 may 1042 1–1 0 unknown
1 unknown 5 may 1467 1–1 0 unknown
2 unknown 5 may 1389 1–1 0 unknown
3 unknown 5 may 579 1–1 0 unknown
4 unknown 5 may 673 2–1 0 unknown
… … … … … … … … …
11154 cellular 8 jul 113 1–1 0 unknown
11157 cellular 20 apr 257 1–1 0 unknown
11158 unknown 16 jun 83 4–1 0 unknown
11159 cellular 19 aug 156 2–1 0 unknown
11161 cellular 9 jul 628 1–1 0 unknown

deposit
0 yes
1 yes
2 yes
3 yes
4 yes
… …
11154 no
11157 no
11158 no
11159 no
11161 no
[7705 rows x 17 columns]
```

Вывод

Как можно заметить выше, pdays = −1 означает, что люди впервые учавствуют в кампании по выдаче кредитов и у них нет предыдущего контакта, поэтому предыдущий контакт == 0. Удалим этим записи.

previous (количество контактов ДО этой кампании)

```
df[df['previous']>30]
age job marital education default
balance housing loan \
2013 46 blue-collar married primary no
1085 yes yes
3677 37 technician married secondary no
432 yes no
6274 27 blue-collar married secondary no
821 yes yes
8713 35 technician single secondary no
4645 yes no
10121 27 student single secondary no 91
no no

contact day month duration campaign
pdays previous poutcome \
2013 cellular 15 may 523 2 353 58 other
3677 cellular 6 jul 386 3 776 55 failure
6274 unknown 16 sep 23 1 778 41 other
8713 cellular 11 jan 502 3 270 40 other
10121 telephone 4 dec 157 6 95 37 other

deposit
2013 yes
```

```
3677 yes
6274 no
8713 no
10121 no
```

Удалим все выбросы в этой функции (см. рис. 6).

```
df.drop(df[df['previous']>30].index,
axis=0, inplace=True)
plt.hist(x=»campaign», data=df,
color='green')
plt.show()
```

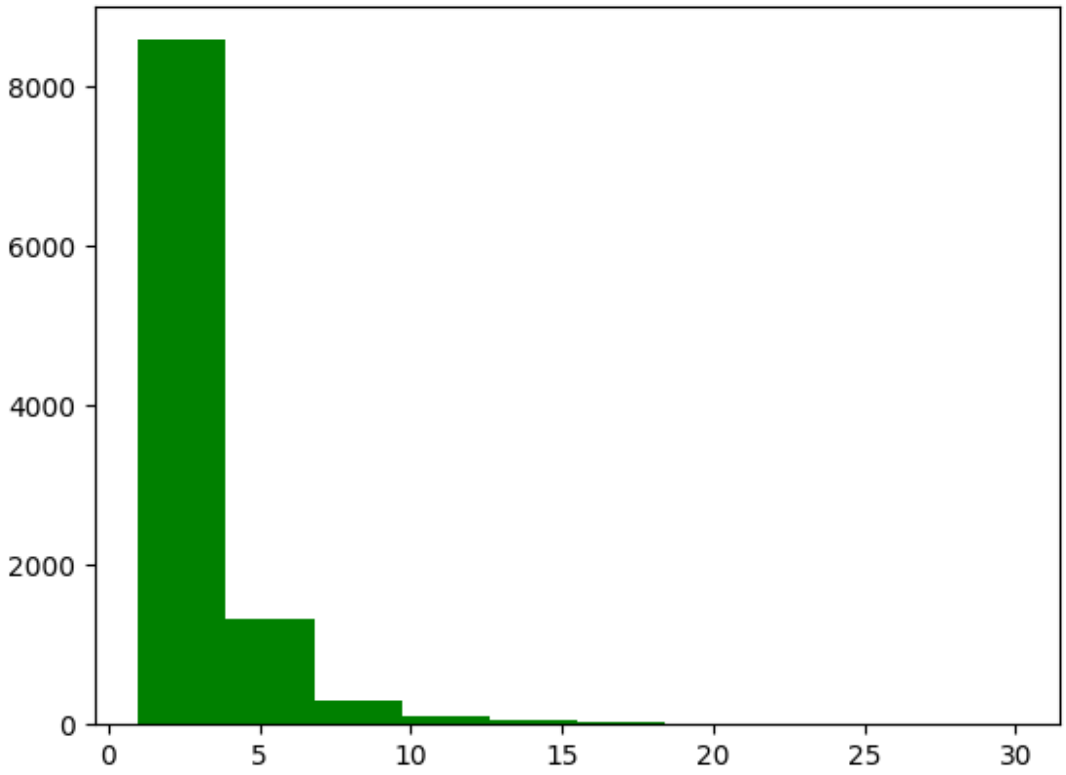

Рис. 6. Гистограмма распределения количества контактов в маркетинговой кампании (составлено авторами)

```
df1 = df.copy()
```

Вывод

Продолжительность последнего контакта — это, то, что больше всего влияет на то, возьмет ли человек кредит или нет.

Сама кампания должна быть как можно меньше дней и у человека должно быть как можно больше денег на счету.

Карта корреляции

Карта корреляции представляет собой инструмент для анализа взаимосвязей между числовыми переменными набора данных. В данной статье была использована корреляция Пирсона, которая позволяет оценить степень линейной зависимости между парами признаков. На тепловой карте визуализированы коэффициенты корреляции, где значения варьируются от–1 до 1: положительное значение указывает на прямую зависимость между переменными, а отрицательное — на обратную.

Результаты показали, что наиболее значимая корреляция наблюдается между переменными duration (длительность разговора) и целевым признаком deposit. Это означает, что продолжительность последнего контакта напрямую влияет на вероятность принятия кредитного предложения. Чем дольше длится разговор, тем выше вероятность положитель-

ного ответа. Остальные признаки, такие как balance (баланс на счету) и campaign (количество контактов в текущей кампании), показали слабую или умеренную корреляцию с целевым признаком.

Важно отметить, что слабая корреляция большинства переменных не означает их бесполезности для модели. Взаимодействие признаков и нелинейные зависимости могут быть учтены с помощью более сложных методов машинного обучения, таких как ансамблевые модели. Таким образом, карта корреляции позволяет выявить сильные линейные связи и служит отправной точкой для дальнейшего анализа данных и выбора ключевых признаков для построения моделей (см. рис. 7).

```
#Используем корреляцию Пирсона, которой измеряют силу линейной зависимости.
df_num = df[['age', 'balance', 'day', 'duration', 'campaign', 'previous']]
plt.figure(figsize=(10,7))
sns.heatmap(data=df_num.corr(), annot=True)
plt.show()
```

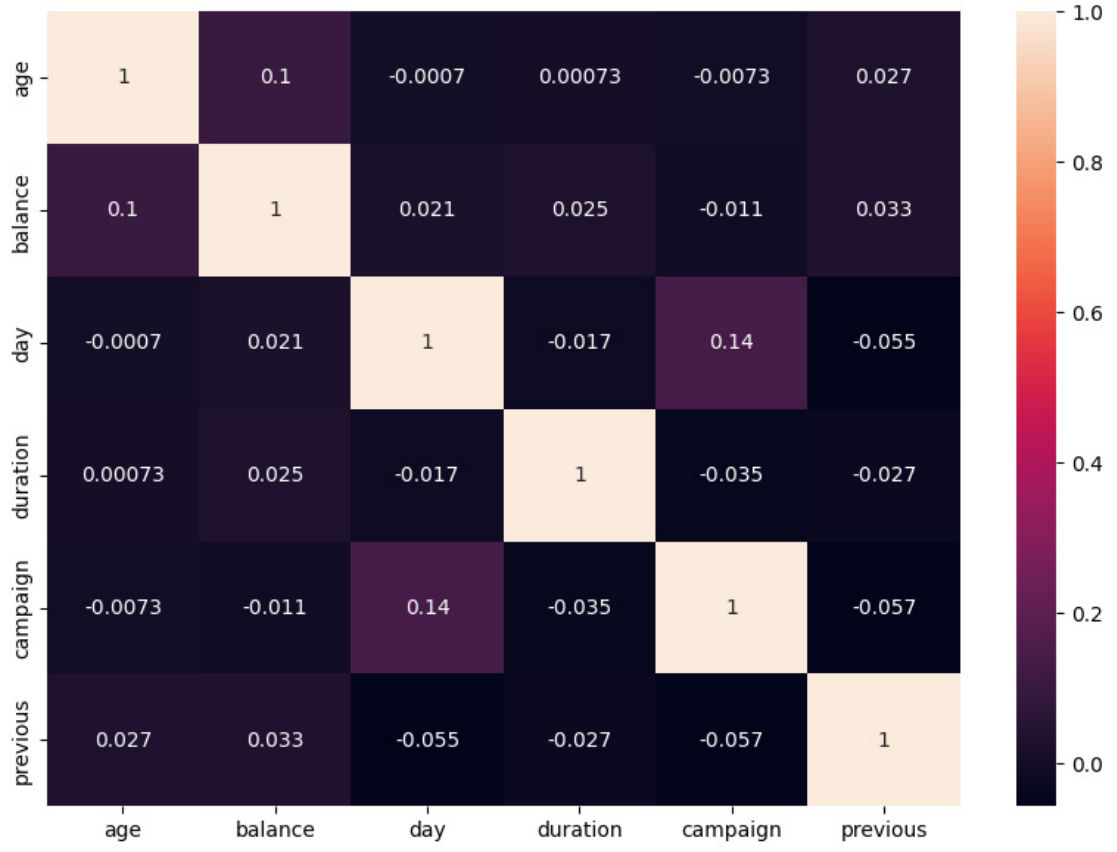

Рис. 7. Корреляционная карта между числовыми признаками, влияющими на выдачу кредита (составлено авторами)

Кодирование категорий

Проведем кодировании категорий. Категориальная переменная — это та, которая имеет два или более значения.

Существует два типа категориальных переменных:

1) номинальные

2) порядковые

Номинальные переменные не имеют внутреннего порядка. Например, пол — это категориальная переменная, имеющая две категории без внутренней упорядоченности. Порядковая категория же имеет четкую структуру.

```
#создаем словарь для бинарного кодирования
dic = {«yes»:1,»no»:0}
```

```
from sklearn.preprocessing import
LabelEncoder
dic = {«yes»:1,»no»:0}
lst = [«deposit»,»loan»,»default»,»hous
ing»]
for i in lst:
df[i] = df[i].map(dic)
# Порядковое кодирование
l=[‘month’,»contact»,»poutcome»]
for i in l:
le=LabelEncoder()
df[i]=le.fit_transform(df[i].values)

df = pd.get_dummies(df, columns =
[‘job’,’marital’,’education’])
df
age default balance housing loan contact
day month duration \
0 59 0 2343 1 0 2 5 8 1042
1 56 0 45 0 0 2 5 8 1467
2 41 0 1270 1 0 2 5 8 1389
3 55 0 2476 1 0 2 5 8 579
4 54 0 184 0 0 2 5 8 673
… … … … … … … … … …
11157 33 0 1 1 0 0 20 0 257
11158 39 0 733 0 0 2 16 6 83
11159 32 0 29 0 0 0 19 1 156
11160 43 0 0 0 1 0 8 8 9
11161 34 0 0 0 0 0 9 5 628

campaign … job_technician job_unemployed
job_unknown \
0 1 … 0 0 0
1 1 … 0 0 0
2 1 … 1 0 0
3 1 … 0 0 0
4 2 … 0 0 0
… … … … … …
11157 1 … 0 0 0
11158 4 … 0 0 0
11159 2 … 1 0 0
11160 2 … 1 0 0
11161 1 … 1 0 0

marital_divorced marital_married
marital_single education_primary \
0 0 1 0 0
1 0 1 0 0
2 0 1 0 0
3 0 1 0 0
4 0 1 0 0
… … … … …
11157 0 0 1 1
11158 0 1 0 0
11159 0 0 1 0
11160 0 1 0 0
11161 0 1 0 0

education_secondary education_tertiary
education_unknown
0 1 0 0
1 1 0 0
2 1 0 0
3 1 0 0
4 0 1 0
… … … …
11157 0 0 0
11158 1 0 0
11159 1 0 0
11160 1 0 0
11161 1 0 0
[10449 rows x 33 columns]
```

Далее необходимо внести поправку в индесы.

```
df=df.reset_index()
df.drop(‘index’, axis=1, inplace=True)
df

age default balance housing loan contact
day month duration \
0 59 0 2343 1 0 2 5 8 1042
1 56 0 45 0 0 2 5 8 1467
2 41 0 1270 1 0 2 5 8 1389
3 55 0 2476 1 0 2 5 8 579
4 54 0 184 0 0 2 5 8 673
… … … … … … … … … …
10444 33 0 1 1 0 0 20 0 257
10445 39 0 733 0 0 2 16 6 83
10446 32 0 29 0 0 0 19 1 156
10447 43 0 0 0 1 0 8 8 9
10448 34 0 0 0 0 0 9 5 628

campaign … job_technician job_unemployed
job_unknown \
0 1 … 0 0 0
```

```
1 1 … 0 0 0
2 1 … 1 0 0
3 1 … 0 0 0
4 2 … 0 0 0
… … … … … …
10444 1 … 0 0 0
10445 4 … 0 0 0
10446 2 … 1 0 0
10447 2 … 1 0 0
10448 1 … 1 0 0

marital_divorced marital_married
marital_single education_primary \
0 0 1 0 0
1 0 1 0 0
2 0 1 0 0
3 0 1 0 0
4 0 1 0 0
… … … … …
10444 0 0 1 1
10445 0 1 0 0
10446 0 0 1 0
10447 0 1 0 0
10448 0 1 0 0

education_secondary education_tertiary
education_unknown
0 1 0 0
1 1 0 0
2 1 0 0
3 1 0 0
4 0 1 0
… … … …
10444 0 0 0
10445 1 0 0
10446 1 0 0
10447 1 0 0
10448 1 0 0
[10449 rows x 33 columns]
```

Предварительная обработка данных

Сначала импортируем библиотеки.

```
from sklearn.pipeline import make_
pipeline
from sklearn.model_selection import
StratifiedShuffleSplit, StratifiedKFold
from sklearn.model_selection import
cross_val_score
from sklearn.model_selection import
GridSearchCV
```

Затем разделим входные и исходящие особенности: X — независимые переменные (входные функции) Y — зависимые переменные (целевая или исходящая переменная).

```
X = df.drop('deposit', axis=1)
Y = df['deposit']
Y
```

```
0 1
1 1
2 1
3 1
4 1
..
10444 0
10445 0
10446 0
10447 0
10448 0
Name: deposit, Length: 10449, dtype:
int64
```

StratifiedShuffleSplit — чтобы сделать равным соотношение категорий для обучающего и тестового набора данных.

StratifiedShuffleSplit представляет собой комбинацию ShuffleSplit и StratifiedKFold. Используя StratifiedShuffleSplit, пропорция распределения меток классов почти одинакова между обучающим и тестовым набором данных.

Основное различие между StratifiedShuffleSplit и StratifiedKFold (shuffle = True) заключается в том, что в StratifiedKFold набор данных перемешивается только один раз в начале, а затем разбивается на указанное количество сгибов. Это отбрасывает любые шансы перекрытия наборов обучающих тестов.

Однако в StratifiedShuffleSplit данные перемешиваются каждый раз перед выполнением разделения, и поэтому существует большая вероятность того, что между наборами обучающих тестов может возникнуть перекрытие.

```
sss = StratifiedShuffleSplit(n_splits=1,
```

```
test_size=0.3, random_state=1)
for train_index, test_index in sss.
split(X, Y):
train_df = df.loc[train_index]
test_df = df.loc[test_index]
```

Соотношение категорий для набора данных обучения и тестирования:

```
print(«Соотношения для тренировочных
данные»)
print(train_df[‘deposit’].value_
counts()/train_df.shape[0])
print()
print(«Соотношение для тестовых данных»)
print(test_df[‘deposit’].value_counts()/
test_df.shape[0])
```

```
Соотношения для тренировочных данные
0 0.51504
1 0.48496
Name: deposit, dtype: float64
```

```
Соотношение для тестовых данных
0 0.515152
1 0.484848
Name: deposit, dtype: float64
```

Создание набора данных для обучения и тестирования

```
X_train = train_df.drop(«deposit»,
axis=1)
Y_train = train_df[‘deposit’]
X_test = test_df.drop(«deposit», axis=1)
Y_test = test_df[‘deposit’]
X_train
```

```
age default balance housing loan contact
day month duration \
5461 28 0 674 1 0 1 14 8 921
4220 36 0 324 1 1 0 16 5 830
5530 56 0 1480 1 1 0 5 3 576
4249 31 0 26965 0 0 0 21 0 654
9514 30 0 177 1 0 0 9 0 62
… … … … … … … … … …
8100 34 0 425 1 0 0 16 5 1389
4223 27 0 11862 0 0 0 25 9 285
343 26 0 551 0 0 0 8 5 531
4449 41 0 5517 1 0 0 10 5 584
2983 76 0 2223 0 0 1 4 3 429
```

```
campaign … job_technician job_unemployed
job_unknown \
5461 4 … 0 0 0
4220 1 … 0 0 0
5530 1 … 1 0 0
4249 2 … 0 0 0
9514 2 … 0 0 0
… … … … … …
8100 7 … 0 0 0
4223 2 … 1 0 0
343 1 … 0 0 0
4449 1 … 0 0 0
2983 1 … 0 0 0
```

```
marital_divorced marital_married
marital_single education_primary \
5461 0 1 0 1
4220 0 1 0 0
5530 1 0 0 0
4249 0 0 1 1
9514 0 1 0 0
… … … … …
8100 0 1 0 1
4223 0 0 1 0
343 0 0 1 0
4449 0 1 0 0
2983 0 1 0 1
```

```
education_secondary education_tertiary
education_unknown
5461 0 0 0
4220 1 0 0
5530 1 0 0
4249 0 0 0
9514 1 0 0
… … … …
8100 0 0 0
4223 0 1 0
343 1 0 0
4449 1 0 0
2983 0 0 0
[7314 rows x 32 columns]
```

Особенности масштабирования

Масштабирование признаков — это метод нормализации/стандартизации независимых признаков, присутствующих в наборе данных, в фиксированном диапазоне.

Есть некоторые алгоритмы, такие как логистическая регрессия, KNN, SVM, которые требуют масштабирования данных для максимальной точности.

Зачем масштабировать функции? Некоторые алгоритмы машинного обучения чувствительны к входящим данным, они работают с формулами расстояния и используют градиентный спуск в качестве оптимизатора.

Наличие значений в одних и тех же масштабах помогает градиентному спуску плавно достигать глобальных минимумов. Например, логистическая регрессия, метод опорных векторов, K- ближайших соседей, K-средние и так далее.

```
from sklearn.preprocessing import
StandardScaler
ss = StandardScaler()
X_train_s = ss.fit_transform(X_train)
X_test_s = ss.transform(X_test)
```

Заключение

В результате исследования был проведен детальный анализ факторов, влияющих на процесс принятия решения о выдаче кредитов с использованием методов машинного обучения. На основе имеющихся данных было выявлено, что наиболее значимыми переменными являются длительность последнего контакта, способ коммуникации и финансовое положение клиента. Длительность разговора оказалась ключевым фактором, напрямую влияющим на вероятность положительного ответа, при этом мобильная связь продемонстрировала преимущество перед другими каналами коммуникации. Высокий баланс счета и успешный опыт предыдущих кампаний также способствовали повышению вероятности одобрения предложения.

Одним из наиболее значимых факторов, определяющих успешное принятие кредитного предложения, является длительность последнего контакта с клиентом. Анализ показал, что чем дольше длился разговор, тем выше вероятность положительного ответа. Это можно объяснить тем, что длительные разговоры позволяют сотрудникам банка предоставить клиенту более полную информацию о продукте, устранить возражения и убедить его в преимуществах предложения. Таким образом, длительность контакта становится не просто показателем активности взаимодействия, но и важным фактором, отражающим эффективность коммуникации.

Еще одним значимым фактором стал способ контакта. Клиенты, с которыми связывались по мобильному телефону, значительно чаще принимали кредитное предложение по сравнению с теми, с кем связывались по другим каналам связи, таким как стационарные телефоны или неизвестные способы связи. Это указывает на важность мобильности и доступности клиентов, что делает канал коммуникации критически важным элементом маркетинговых кампаний.

Таким образом, сочетание этих факторов позволяет нам строить более точные и эффективные модели для прогнозирования поведения клиентов и оптимизации кредитных кампаний, что в конечном итоге может повысить прибыльность и успешность банковских предложений.

Литература

1. Далингер К. Е., Айграшева А. А. (2021). Анализ текущего состояния льготного ипотечного кредитования в Российской Федерации // Вопросы студенческой науки, выпуск № 10 (62)// Электронный ресурс (октябрь 2021 г.) //URL: https://sciff.ru/wp-content/uploads/2021/11/Sciff_10_62.pdf (дата обращение 5 июля 2023 года).
2. Мыцык И. В., Кузнецова Е. В. (2023). Современные проблемы ипотечного кредитования в России и анализ перспектив их

решения // Международный научный журнал «Инновационная наука» № 5–1// Электронный ресурс, 2023//URL: https://aeterna-ufa.ru/sbornik/IN-2023–05–1.pdf (дата обращение 5 июля 2023 года).
3. Предет Р. В. (2023). Анализ банковского кредитования субъектов малого и среднего бизнеса // Форум молодых ученых, 3(79) // Электронный ресурс, 2023//URL: https://www.forum-nauka.ru/_files/ugd/b06fdc_ded5cc7854e341879805bc7e3d415403.pdf?index=true (дата обращение 5 июля 2023 года).
4. Мишина М. Ю., Зверев А. В. (2021). Статистический анализ состояния ипотечного кредитования в Российской Федерации // Статистический анализ социально-экономического развития субъектов Российской Федерации // Сборник научных трудов по материалам VIII Международной научно-практической конференции. Брянск// Издательство: Федеральное государственное бюджетное образовательное учреждение высшего образования «Брянский государственный инженерно-технологический университет» (Брянск).

Methods of data analysis and preparation for modeling in the credit sphere

Demidov A. D., Bulgakov A. L.,
Lomonosov Moscow State University; Plekhanov Russian University of Economics,
Moscow Institute of Modern Academic Education

The article considers the process of data analysis and preparation for modeling for forecasting the creditworthiness of clients of banking institutions. An overview of modern approaches to data processing is presented, including identifying and eliminating anomalies, class balancing and visualization of key characteristics. A detailed analysis of the data set is conducted, including the study of categorical and numerical features, as well as their impact on the probability of obtaining a loan. The main stages of data preparation for machine learning are described, such as cleaning, transformation and creation of new features. The paper emphasizes the importance of high-quality data preprocessing to improve the accuracy of forecasts and the efficiency of model building. The results of the study form the basis for the subsequent application of machine learning algorithms and analysis of their performance in lending problems.

Keywords: credit scoring systems, data analysis, lending.

References:

1. Dalinger K. E., Aigrasheva A. A. (2021). Analysis of the current state of preferential mortgage lending in the Russian Federation // Issues of student science, issue No. 10 (62) // Electronic resource (October 2021) // URL: https://sciff.ru/wp-content/uploads/2021/11/Sciff_10_62.pdf (date accessed July 5, 2023).
2. Mytsyk I. V., Kuznetsova E. V. (2023). Current problems of mortgage lending in Russia and analysis of prospects for their solution // International scientific journal «Innovation Science» No. 5–1 // Electronic resource, 2023 // URL: https://aeterna-ufa.ru/sbornik/IN-2023–05–1.pdf (date accessed July 5, 2023).
3. Predet R. V. (2023). Analysis of bank lending to small and medium-sized businesses // Forum of young scientists, 3 (79) // Electronic resource, 2023 // URL: https://www.forum-nauka.ru/_files/ugd/b06fdc_ded5cc7854e341879805bc7e3d415403.pdf?index=true (date accessed July 5, 2023).
4. Mishina M. Yu., Zverev A. V. (2021). Statistical analysis of the state of mortgage lending in the Russian Federation // Statistical analysis of the socio-economic development of the constituent entities of the Russian Federation // Collection of scientific papers based on the materials of the VIII International Scientific and Practical Conference. Bryansk // Publisher: Federal State Budgetary Educational Institution of Higher Education «Bryansk State University of Engineering and Technology» (Bryansk).