

Національний технічний університет України «КПІ ім. Ігоря Сікорського»  
Факультет Інформатики та Обчислювальної Техніки



Кафедра інформаційних систем та технологій

## Лабораторна робота №8 з дисципліни «Вступ до технології Data Science»

на тему

«МАКЕТ CRM СИСТЕМИ SCORING – АНАЛІЗУ  
(міні проекти в банківській сфері аналізу даних)»

Виконала:  
студентка групи ІС-12  
Павлова Софія

Перевірив:  
Баран Д. Р.

# 1. Постановка задачі

## Мета роботи:

Дослідити виявити та узагальнити особливості реалізації проектного практикуму в галузі кредитного scoring-у.

## Завдання:

Відділ мікрокредитування банківської установи замовив розробку Backend компоненту SRM банківської системи.

### **Компонента Backend повинна забезпечити:**

1. Побудову скорінгової оцінку;
2. Кластеризацію заявників за бінарної оцінкою.

### **Вихідні дані для опрацювання задачі представлені у формі 2-х файлів:**

1. *sample\_data.xlsx* – файл реальних даних про позичальників та проміжних параметрів банківських індикаторів;
2. *data\_description.xlsx* – файл пояснень структури скорінгової таблиці та тлумачення Індикаторів.

Розробити програмний скрипт, що реалізує функціонал за обраним рівням складності:

## Завдання III рівня:

Розробити програмний скрипт, що реалізує: скорінговий аналіз позичальників за даними скорінгової карти; виявлення шахрайства та фальсифікації даних – за методиками / алгоритмами, що є R&D – новацією.

## 2. Виконання

### 2.1. Постановка задачі

Для початку розберімося з поняттями:

**Кредитний скоринг** – це автоматична бальна система оцінки позичальника. Кожен клієнт банку проходить анкетування – залишає про себе докладні дані. Будь-яка його характеристика має своє значення в балах. Після перевірки достовірності цих даних і підсумовування набраних балів ухвалюється рішення щодо платоспроможності потенційного позичальника і, виходячи з цього, про видачу або не видачу кредиту.

У основі кредитного скорингу лежить припущення, що люди зі схожими соціальними показниками поведуться однаково. Простіше кажучи, якщо у банку вже є певна кількість недисциплінованих клієнтів з певними характеристиками, то він буде думати, що платіжна дисципліна інших з подібними реквізитами не буде кращою. Виходячи з цього, будуються скорингові карти, на основі яких визначаються значення скоринг-балів.

**Завдання для скорингу:** Для реалізації скорингу використаємо метод «скорингової моделі для надання кредиту з використанням оцінки параметрів клієнта». Визначимо різні показники для кожного клієнта, призначимо їм ваги, розраховуємо загальний бал та приймемо рішення про надання кредиту на основі цього балу. Використаємо для цього **евристичні функції**.

Відповідно до завдання реалізуємо **скоринговий аналіз за алгоритмом**:

#### 1. Підготовка даних

1.1. Парсинг файлу вхідних даних

1.2. Аналіз структури вхідних даних

1.3. Первинне формування скорингової таблиці

## 1.4. Очищення даних

1.4.1. Аналіз перетину скорингових індикаторів та сегменту вхідних даних

1.4.2. Формування DataFrame з даних з урахуванням відсутніх індикаторів скорингової таблиці

1.4.3. Очищення скорингової таблиці від пропусків

## 2. Формування скорингової моделі

2.1. Парсинг файлу індикаторів

2.2. Оцінка кількості балів для різних параметрів клієнта з використанням евристичних функцій

2.3. Розрахунок загального балу

2.4. Прийняття рішення

**Скоринг шахрайства** – це вид скоринга, який є складною системою виявлення будь-яких невідповідностей або навпаки збігів, які також виявляються за допомогою перехресних перевірок. Його мета – виявити все, що може викликати підозри.

**Завдання для детекції шахрайства:** Шахраїв будемо визначати за допомогою пошуку аномалій у вхідних даних. Для цього задіємо знання, отримані з попередніх лабораторних робіт.

## 2.2. Вхідні дані

Маємо два файли вхідних даних.

	AU	AV	AW	AX	AY	AZ	BA	BB	BC	BD	BE	BF	BG	BH	BI	BJ	BK	BL	BM	BN	BO
1	amount_to_pay	total_paid	current_interest	current_overdue	current_rest_amount	current_overdue	overdue_extensor	extensor	payment_date	closed_at	product	credit_pouser_id	face_id	prolongat	prolongat	wizard	ty	step	created_at		
2	0	4620	1620	0	0	0	0	0	0	02.03.2021	01.03.2021 16:17	14	31	253430	NULL	NULL	NULL	2	3	#####	
3	0	1120	120	0	0	0	0	0	0	07.02.2021	07.02.2021 15:30	12	38	109618	NULL	NULL	NULL	2	3	#####	
4	1088	1183	145	38	0	0	0	2	0	03.02.2021	05.02.2021 18:26	14	39	289130	NULL	NULL	NULL	2	3	#####	
5	0	2168,4	568,4	0	0	0	0	0	0	02.03.2021	18.02.2021 18:42	18	38	141625	226701	NULL	NULL	2	5	7	#####
6	3402,5	11290	5322,5	3467,5	0	0	0	73	0	19.04.2021	23.05.2021 18:29	14	39	204249	NULL	1	20	5	7	#####	
7	0	1472,48	443,8	28,68	0	0	0	2	0	03.04.2021	11.03.2021 15:22	14	31	122083	NULL	1	30	5	7	#####	
8	2028,6	3435,2	1407,2	728	0	0	0	28	0	02.03.2021	30.03.2021 10:24	18	38	225205	NULL	NULL	NULL	2	3	#####	
9	0	2506	506	0	0	0	0	0	0	15.02.2021	13.02.2021 11:02	14	39	209131	NULL	NULL	NULL	5	7	#####	
10	0	2847	347	0	0	0	0	0	0	12.02.2021	08.02.2021 21:13	14	31	276402	NULL	NULL	NULL	2	3	#####	
11	0	1120	120	0	0	0	0	0	0	07.02.2021	03.02.2021 9:55	18	38	1777	NULL	NULL	NULL	2	3	#####	
12	0	2476	476	0	0	0	0	0	0	02.03.2021	12.02.2021 4:14	14	31	223487	NULL	NULL	NULL	2	3	#####	
13	0	2161,2	705,2	56	0	0	0	2	0	07.03.2021	26.02.2021 23:17	14	31	250637	NULL	1	30	2	3	#####	
14	3927,5	8630	3802,5	2327,5	0	0	0	49	0	03.03.2021	21.04.2021 15:52	14	39	289142	NULL	NULL	NULL	5	7	#####	
15	0	2879,4	879,4	0	0	0	0	0	0	03.03.2021	02.03.2021 10:39	14	31	116802	NULL	NULL	NULL	2	3	#####	
16	2215,55	2406,5	807,35	99,15	0	0	0	3	0	03.03.2021	06.03.2021 4:25	14	31	241322	NULL	NULL	NULL	2	3	#####	
17	0	1318	318	0	0	0	0	0	0	03.03.2021	17.02.2021 5:42	14	31	24870	NULL	NULL	NULL	2	3	#####	
18	0	876,8	76,8	0	0	0	0	0	0	04.02.2021	NULL	14	31	236645	NULL	NULL	NULL	2	3	#####	
19	1422,8	1528,6	287,6	41	0	0	0	2	0	10.02.2021	12.02.2021 16:11	14	31	281395	NULL	NULL	NULL	2	3	#####	
20	4620	6156	2376	780	0	0	0	13	0	03.03.2021	16.03.2021 13:19	14	31	235734	NULL	NULL	NULL	2	3	#####	
21	0	2245,7	295,7	0	0	0	0	0	0	03.03.2021	08.02.2021 4:00	14	31	106816	NULL	NULL	NULL	2	3	#####	
22	0	1867,4	167,4	0	0	0	0	0	0	10.02.2021	05.02.2021 2:20	14	31	86072	194019	NULL	NULL	2	3	#####	
23	0	4145	1095	0	0	0	0	0	0	21.02.2021	21.02.2021 18:06	14	31	153435	237604	NULL	NULL	2	3	#####	
24	0	3459,5	1874	85,5	0	0	0	3	0	05.05.2021	06.04.2021 21:09	14	39	289167	NULL	2	60	5	7	#####	
25	1806,5	2006	420,5	85,5	0	0	0	3	0	11.02.2021	14.02.2021 14:46	14	39	286897	NULL	NULL	NULL	5	7	#####	
26	0	1464,7	464,7	0	0	0	0	0	0	03.03.2021	03.03.2021 13:21	14	39	284922	NULL	NULL	NULL	5	7	#####	
27	0	3648,5	648,5	0	0	0	0	0	0	17.02.2021	17.02.2021 23:13	14	31	263804	NULL	NULL	NULL	2	3	#####	
28	1739	2333	536	297	0	0	0	10	0	09.02.2021	19.02.2021 20:53	14	31	272420	NULL	NULL	NULL	2	3	#####	
29	0	568	68	0	0	0	0	0	0	03.03.2021	01.02.2021 23:29	14	31	254736	NULL	NULL	NULL	2	3	#####	
30	0	2126	126	0	0	0	0	0	0	15.02.2021	02.02.2021 13:01	14	39	289188	NULL	NULL	NULL	5	7	#####	

Рисунок 1 – Вхідний файл *sample\_data.xlsx*

Таблицю, що має дані дані, які клієнти ввели про себе на сайті.

	B	C	D	E	F
1	Description of information	Filling	At the time of the request	Place of definition	Note
2	Ідентифікатор заявки	Обов'язково	Відомо	Визначається системою обліку заявок на момент реєстрації	
3	Сума кредиту в заявці	Обов'язково	Відомо	Вказує позичальник	
4	Термін погашення кредиту в заявці	Обов'язково	Відомо	Вказує позичальник	
5	Дата і час прийняття заявки	Обов'язково	Відомо	Визначається при подачі заявки	Локальний час на пристрої користувача або час прийому заявок?
6				Визначається браузером при подачі заявки	Завжди UA
7	ID мети використання кредиту	Обов'язково	Відомо		Завжди 999 – 999 це клієнт вписує саме призначення, раніше можна було вибирати зі списку
8	Мета використання кредиту	Обов'язково	Відомо	Вказує позичальник	
9	IP клієнта, з якого надійшов запит	Обов'язково	Відомо	Визначається браузером при подачі заявки	
10	Технічна інформація про пристрій клієнта	Обов'язково	Відомо	Визначається браузером при подачі заявки	
11	Сайт з якого користувач прийшов на сторінку заявки	Відомо	Відомо	Визначається браузером при подачі заявки	
12	не використовується				Завжди пусто
13	Стать позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
14	Дата народження позичальника	Обов'язково	Відомо	Вказує позичальник	Іноді заповнюється не вірно
15	Сімейний статус позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
16	Кількість дітей у позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
17	Рівень освіти позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
18	Номер паспорта	Обов'язково	Відомо	Вказує позичальник	Повинен бути заповнений
19	Дата видачі паспорта	Обов'язково	Відомо	Вказує позичальник	Повинна бути заповнена вірно
20	Кім виданий паспорт	Відомо	Відомо	Вказує позичальник	
21	ИНН позичальника	Обов'язково	Відомо	Вказує позичальник	Повинен бути заповнений
22	email позичальника	Обов'язково	Відомо	Вказує позичальник	Повинен бути заповнений
23	Мобільний телефон позичальника	Обов'язково	Відомо	Вказує позичальник	Повинен бути заповнений
24	Посилання на профілі в соціальних мережах	Відомо	Відомо	Вказує позичальник	Будет ли информация на момент заявки?
25	Адреса реєстрації: область	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
26	Адреса реєстрації: поштовий індекс	Відомо	Відомо	Вказує позичальник	
27	Адреса реєстрації: місто	Відомо	Відомо	Вказує позичальник	
28	Адреса реєстрації: тип вулиці	Відомо	Відомо	Вказує позичальник	
29	Адреса реєстрації: вулиця	Відомо	Відомо	Вказує позичальник	
30	Адреса реєстрації: номер дома	Відомо	Відомо	Вказує позичальник	

Рисунок 2 – Вхідний файл *data\_description.xlsx*

Та таблицю, що описує природу показників скорингової карти.

## 2.3. Скоринговий аналіз

### Парсинг файлу вхідних даних

Для початку виконаємо парсинг файлу *sample\_data.xlsx*.

#### Лістинг коду:

```
import pandas as pd
import numpy as np

# Парсинг файлу вхідних даних
d_sample_data = pd.read_excel('sample_data.xlsx', parse_dates=['birth_date'])
print('\nСтруктура скорингової карти індикаторів')
print(d_sample_data)
Title d_sample_data = d_sample_data.columns
```

#### Результат:

```
Реальні дані про позичальників
  Application  loan_amount  ... step      created_at
0           1         3000  ...   3  2021-02-01 00:20:01
1           2         1000  ...   3  2021-02-01 00:20:04
2           3         1000  ...   7  2021-02-01 00:22:13
3           4         1600  ...   3  2021-02-01 00:22:15
4           5         2500  ...   7  2021-02-01 00:23:03
..          ...          ...  ...  ...          ...
494         495         1000  ...   7  2021-02-03 00:55:41
495         496          500  ...   7  2021-02-03 01:23:54
496         497         2500  ...   7  2021-02-03 01:43:13
497         498         1050  ...   3  2021-02-03 01:51:31
498         499         1300  ...   3  2021-02-03 01:54:18

[499 rows x 67 columns]
```

Рисунок 3 – Реальні дані позичальників

Бачимо, що фрагмент структури вхідних даних має інформацію про 499 клієнтів по 67 індикаторах.

Запарсимо другий файл: *data\_description.xlsx*.

### Лістинг коду:

```
# Парсинг файлу пояснень параметрів
d_data_description = pd.read_excel('data_description.xlsx')
print('\nСтруктура скорингової карти індикаторів')
print(d_data_description)
```

### Результат:

```
Структура скорингової карти індикаторів
   Field_in_data  ... Note
0             id  ...  NaN
1    loan_amount  ...  NaN
2    loan_days   ...  NaN
3    applied_at  ...  Локальний час на пристрої користувача або час ...
4    language   ...  Завжди UA
..            ...  ...  ...
115 cardholder_name  ...  NaN
116         face_id  ...  NaN
117          step   ...  NaN
118    created_at  ...  Локальний час на пристрої користувача або час ...
119    updated_at  ...  Локальний час на пристрої користувача або час ...

[120 rows x 6 columns]
```

Рисунок 4 – Структура скорингової карти індикаторів

Бачимо, що заявлена структура скорингової карти має 120 індикаторів. Отримуємо розбіжність розмірностей вхідних даних. Для усунення цієї розбіжності доведеться штучно стискати дані.

### Аналіз структури вхідних даних

Почнемо підготовку до штучного стискання даних. Виконаємо аналіз структури вхідного файлу. Виведемо назви цих 67 індикаторів, що являють собою стовпці нашої структури, їх тип та кількість пропущених значень у цих стовпцях.

### Лістинг коду:

```
# Аналіз структури вхідних даних
print('\nНазви стовпців DataFrame')
print>Title_d_sample_data
print('\nТипи даних стовпців DataFrame')
print(d_sample_data.dtypes)
print('\nПропущені значення стовпців (суми)')
print(d_sample_data.isnull().sum())
```

## Результат:

```
Назви стовпців DataFrame
Index(['Application', 'loan_amount', 'loan_days', 'applied_at', 'gender_id',
      'Unnamed: 5', 'birth_date', 'Marital status', 'children_count_id',
      'education_id', 'fact_addr_owner_type_id', 'fact_addr_start_date',
      'has_immovables', 'has_movables', 'employment_type_id', 'position_id',
      'organization_type_id', 'organization_branch_id', 'employees_count_id',
      'employment_date', 'seniority_years', 'has_prior_employment',
      'prior_employment_start_date', 'prior_employment_end_date',
      'monthly_income', 'income_frequency_id', 'income_frequency_other',
      'income_source_id', 'monthly_expenses', 'other_loans_has_closed',
      'other_loans_active', 'other_loans_about_current',
      'other_loans_about_monthly', 'OK*', 'loan_closed', 'loan_overdue',
      'product_id', 'product_dpr', 'product_amount_from', 'product_amount_to',
      'product_overdue_dpr', 'product_interest_min',
      'product_overdue_start_day', 'product_base_amount_limit',
      'amount_limit', 'applied_limit', 'amount_to_pay', 'total_paid',
      'current_interest_amount', 'current_overdue_interest_amount',
      'current_rest_amount', 'current_principal_balance', 'overdue_amount',
      'overdue_days', 'extension_amount', 'extension_days', 'payment_date',
      'closed_at', 'product_profile_id', 'credit_policy_id', 'user_id',
      'face_id', 'prolongation_number', 'prolongation_total_days',
      'wizard_type_id', 'step', 'created_at'],
      dtype='object')
```

Рисунок 5 – Назви стовпців вхідної структури

```
Типи даних стовпців DataFrame
Application          int64
loan_amount          int64
loan_days            int64
applied_at           datetime64[ns]
gender_id            int64
...
prolongation_number  float64
prolongation_total_days float64
wizard_type_id       int64
step                 int64
created_at            datetime64[ns]
Length: 67, dtype: object
```

Рисунок 6 – Типи даних стовпців вхідної структури

Так, як пайон має несувору типізовану структуру, бачимо, що для кожного індикатора різний тип даних у нашій вхідній структурі.



```
Пропущені значення стовпців (суми)
Application          0
loan_amount          0
loan_days            0
applied_at           0
gender_id            0

prolongation_number  436
prolongation_total_days 436
wizard_type_id       0
step                 0
created_at           0
Length: 67, dtype: int64
```

Рисунок 7 – Кількість пропущених значень у стовпцях індикаторів

Деякі індикатори мають значні пропуски в даних. За такої кількості пропусків раціонально відкинути такі значення для збереження точності скорингового аналізу.

### Первинне формування скорингової таблиці

Сформуємо скорингову таблицю з акцентом на первинні дані про клієнта і на суб'єктивні рішення банку. Для цього виберемо з переліку індикаторів параметри «Вказує позичальник» та «Параметри, пов'язані з виданим продуктом».

### Лістинг коду:

```
# Первинне формування скорингової таблиці
# Сегментація ознак клієнта та кредиту
d_segment_data_description_client_bank =
d_data_description[(d_data_description.Place_of_definition == 'Вказує позичальник')
                    |
                    (d_data_description.Place_of_definition == 'параметри, пов'язані з виданим продуктом')]
n_client_bank = d_segment_data_description_client_bank['Place_of_definition'].size
d_segment_data_description_client_bank.index = range(0,
len(d_segment_data_description_client_bank))
print('\nПервинне формування скорингової карти')
print(d segment data description client bank)
```

## Результат:

```
Первинне формування скорингової карти
Field_in_data ... Note
0 loan_amount ... NaN
1 loan_days ... NaN
2 purpose_other ... NaN
3 gender_id ... Потрібна расшифровка id
4 birth_date ... Іноді заповнюється не вірно
.. ... ..
73 product_overdue_start_day ... NaN
74 product_base_amount_limit ... NaN
75 amount_limit ... NaN
76 applied_limit ... NaN
77 amount_to_pay ... NaN
```

[78 rows x 6 columns]

Рисунок 8 – Структура скорингової карти індикаторів

Сформована певинна скорингова карта має вже 78 індикаторів замість 120. Але наша вхідна структура має все ще менше.

## **Аналіз перетину скорингових індикаторів та сегменту вхідних даних**

Перевіримо факт неспівпадиння переліку стовпців даних за колонками та переліку індикаторів скорингової таблиці. Підрахуємо кількість спільних індикаторів скорингової таблиці. І на решті визначимо позиції спільних індикаторів скорингової таблиці за індексом.

## Лістинг коду:

```
# Перевірка наявності індексів клієнта та кредиту з d_data_description в даних
d_sample_data
b = d_segment_data_description_client_bank['Field_in_data']

# Кількість співпадинь
n_columns = d_segment_data_description_client_bank['Field_in_data'].size
j = 0
for i in range(0, n_columns):
    a = d_segment_data_description_client_bank['Field_in_data'][i]
    if set([a]).issubset(d_sample_data.columns):
        j = j + 1
print(' j = ', j)

# Індекси співпадинь
Columns_Flag_True = np.zeros((j))
j = 0
```

```

for i in range(0, n_columns):
    a = d_segment_data_description_client_bank['Field_in_data'][i]
    if set([a]).issubset(d_sample_data.columns):
        Flag = 'Flag_True'
        Columns_Flag_True[j] = i
        j = j + 1
    else:
        Flag = 'Flag_False'
print('Індекси співпадінь', Columns_Flag_True)

```

Сформуємо DataFrame за індексами, які відповідають спільним індикаторам.

### Лістинг коду:

```

# Формування DataFrame даних з урахуванням відсутніх індикаторів скорингової таблиці
d_segment_data_description_client_bank_True =
d_segment_data_description_client_bank.iloc[Columns_Flag_True]
d_segment_data_description_client_bank_True.index = range(0,
len(d_segment_data_description_client_bank_True))
print('\nDataFrame співпадінь')
print( d_segment_data_description_client_bank_True)

```

### Результат:

20	monthly_income	...	NaN
21	income_frequency_id	...	NaN
22	income_frequency_other	...	NaN
23	income_source_id	...	NaN
24	monthly_expenses	...	NaN
25	other_loans_has_closed	...	NaN
26	other_loans_active	...	NaN
27	other_loans_about_current	...	NaN
28	other_loans_about_monthly	...	NaN
29	product_id	...	NaN
30	product_dpr	...	NaN
31	product_amount_from	...	NaN
32	product_amount_to	...	NaN
33	product_interest_min	...	NaN
34	product_base_amount_limit	...	NaN
35	amount_limit	...	NaN
36	applied_limit	...	NaN
37	amount_to_pay	...	NaN

[38 rows x 6 columns]

Рисунок 9 –DataFrame співпадінь

Бачимо, що за співпадіннями ми отримуємо 38 спільних індикаторів. Саме вони будуть у нашій майбутній скоринговій таблиці.

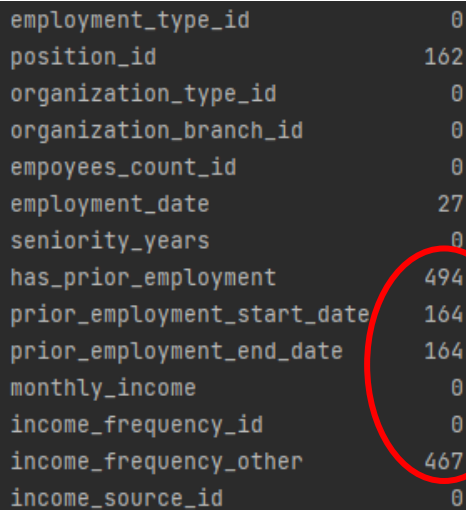
## Очищення скорингової таблиці від пропусків

Визначимо кількість пропусків по кожному індикатору в новому DataFrame.

### Лістинг коду:

```
# Формування сегменту вхідних даних за рейтингом клієнт + банк
b = d_segment_data_description_client_bank_True['Field_in_data']
d_segment_sample_data_client_bank = d_sample_data[b]
print('\nПропуски даних сегменту DataFrame')
print(d_segment_sample_data_client_bank.isnull().sum())
```

### Результат:



employment_type_id	0
position_id	162
organization_type_id	0
organization_branch_id	0
employees_count_id	0
employment_date	27
seniority_years	0
has_prior_employment	494
prior_employment_start_date	164
prior_employment_end_date	164
monthly_income	0
income_frequency_id	0
income_frequency_other	467
income_source_id	0

Рисунок 10 – Пропуски даних нового DataFrame

Бачимо, що для деяких індикаторів все ще забагато втрачених даних. Тому проведемо політику видалення індикаторів, що мають пропуски.

## Формування DataFrame даних з урахуванням відсутніх індикаторів скорингової таблиці

Вилучимо такі індикатори та проведемо контрольну перевірку.

### Лістинг коду:

```
# Очищення індикаторів скорингової таблиці
d_segment_data_description_cleaning = d_segment_data_description_client_bank_True.loc[
    (d_segment_data_description_client_bank_True['Field_in_data'] !=
     'fact addr start date')]
d_segment_data_description_cleaning = d_segment_data_description_cleaning.loc[
    (d_segment_data_description_cleaning['Field_in_data'] != 'position id')]
```

```

d_segment_data_description_cleaning = d_segment_data_description_cleaning.loc[
    (d_segment_data_description_cleaning['Field_in_data'] != 'employment_date')]
d_segment_data_description_cleaning = d_segment_data_description_cleaning.loc[
    (d_segment_data_description_cleaning['Field_in_data'] != 'has_prior_employment')]
d_segment_data_description_cleaning = d_segment_data_description_cleaning.loc[
    (d_segment_data_description_cleaning['Field_in_data'] !=
'prior_employment_start_date')]
d_segment_data_description_cleaning = d_segment_data_description_cleaning.loc[
    (d_segment_data_description_cleaning['Field_in_data'] !=
'prior_employment_end_date')]
d_segment_data_description_cleaning = d_segment_data_description_cleaning.loc[
    (d_segment_data_description_cleaning['Field_in_data'] != 'income_frequency_other')]
d_segment_data_description_cleaning.index = range(0,
len(d_segment_data_description_cleaning))
d_segment_data_description_cleaning.to_excel('d_segment_data_description_cleaning.xlsx')

# Очищення вхідних даних
d_segment_sample_cleaning =
d_segment_sample_data_client_bank.drop(columns=['fact_addr_start_date', 'position_id',
'employment_date',

'has_prior_employment', 'prior_employment_start_date',

'prior_employment_end_date', 'income_frequency_other'])
d_segment_sample_cleaning.index = range(0, len(d_segment_sample_cleaning))
d_segment_sample_cleaning.to_excel('d_segment_sample_cleaning.xlsx')
print('\nКонтроль наявності пропусків даних після очищення на індикаторах')
print(d_segment_sample_cleaning.isnull().sum())
print('\nDataFrame вхідних даних - скорингова карта')
print(d_segment_sample_cleaning)
print('\nDataFrame індикатори скорингу')
print(d_segment_data_description_cleaning)

```

### **Результат:**

other_loans_has_closed	0
other_loans_active	0
other_loans_about_current	0
other_loans_about_monthly	0
product_id	0
product_dpr	0
product_amount_from	0
product_amount_to	0
product_interest_min	0
product_base_amount_limit	0
amount_limit	0
applied_limit	0
amount_to_pay	0

Рисунок 11 – Контроль наявності пропусків після очищення на індикаторах

Бачимо, що нові дані дійсно візуально не мають пропусків. І в результаті отримуємо наступні вхідні дані для подальшого скорингового аналізу.

```
DataFrame вхідних даних - скорингова карта
  loan_amount  loan_days  ...  applied_limit  amount_to_pay
0         3000         30  ...         3000         0.0
1         1000          7  ...         1000         0.0
2         1000          3  ...         1000        1088.0
3         1600         30  ...         1600         0.0
4         2500         18  ...         2500        3402.5
..         ...         ...  ...         ...         ...
494         1000         15  ...         1000         0.0
495          500          3  ...         1500         0.0
496         2500         30  ...         2500         0.0
497         1050         14  ...         3500        1345.7
498         1300          4  ...         5000         0.0

[499 rows x 31 columns]
```

Рисунок 12 – Фінальна скорингова карта

```
DataFrame індикатори скорингу
  Field_in_data  ...  Note
0      loan_amount  ...  NaN
1      loan_days  ...  NaN
2      gender_id  ...  Потрібна расшифровка id
3      birth_date  ...  Іноді заповнюється не вірно
4      children_count_id  ...  Потрібна расшифровка id
5      education_id  ...  Потрібна расшифровка id
6      fact_addr_owner_type_id  ...  Потрібна расшифровка id
7      has_immovables  ...  NaN
8      has_movables  ...  NaN
9      employment_type_id  ...  Потрібна расшифровка id
10     organization_type_id  ...  Потрібна расшифровка id
11     organization_branch_id  ...  Потрібна расшифровка id
12     employees_count_id  ...  Потрібна расшифровка id
13     seniority_years  ...  Повинен бути вказаний вірно
14     monthly_income  ...  NaN
15     income_frequency_id  ...  NaN
16     income_source_id  ...  NaN
17     monthly_expenses  ...  NaN
18     other_loans_has_closed  ...  NaN
19     other_loans_active  ...  NaN
20     other_loans_about_current  ...  NaN
21     other_loans_about_monthly  ...  NaN
22     product_id  ...  NaN
23     product_dpr  ...  NaN
24     product_amount_from  ...  NaN
25     product_amount_to  ...  NaN
26     product_interest_min  ...  NaN
27     product_base_amount_limit  ...  NaN
28     amount_limit  ...  NaN
29     applied_limit  ...  NaN
30     amount_to_pay  ...  NaN

[31 rows x 6 columns]
```

Рисунок 13 – Фінальні ідентифікатори скорингу

Бачимо, що їх розмірності співпадають, а отже можна переходити до наступного етапу.

## Формування скорингової моделі

Оскільки позичальник не вказав склад скорингової карти, таку задачу прийнято вважати некоректною. Виведемо задачу із розряду «некоректної» методом здорового глузду.

Приймемо до уваги **13 ідентифікаторів**: *Сума кредиту, Термін погашення кредиту, Стать, Кількість дітей, Рівень освіти, Наявність нерухомості, Наявність транспорту, Тип зайнятості, Щомісячні доходи, Щомісячні витрати, Присутність погашених позик, Присутність непогашених позик, Загальний розмір непогашених позик.*

Завантажимо дані та почнемо формувати оцінку кількості балів для різних параметрів клієнта. Зімітуємо так звану «відкриту» скорингову карту, де банк надає нам розшифровку скорингового балу для різних ідентифікаторів.

### Лістинг коду:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from pyod.models.knn import KNN
import seaborn as sns

# Оцінка суми кредиту
def evaluate_loan_amount(loan_amount):
    if loan_amount < 1000:
        return 4
    elif 1000 <= loan_amount < 5000:
        return 3
    elif 5000 <= loan_amount < 10000:
        return 2
    else:
        return 1

# Оцінка терміну погашення кредиту
def evaluate_loan_days(loan_days):
    if loan_days < 7:
        return 4
    elif 7 <= loan_days < 30:
        return 3
    elif 30 <= loan_days < 90:
        return 2
```

```
    else:
        return 1

# Оцінка статі
def evaluate_gender(gender_id):
    if gender_id == 1:
        return 2
    else:
        return 1

# Оцінка кількості дітей
def evaluate_children_count(children_count_id):
    if children_count_id == 0:
        return 1
    elif 1 <= children_count_id < 2:
        return 2
    elif 3 <= children_count_id < 4:
        return 3
    else:
        return 4

# Оцінка рівня освіти
def evaluate_education(education_id):
    if education_id == 1:
        return 1
    elif 2 <= education_id < 3:
        return 2
    elif 4 <= education_id < 5:
        return 3
    else:
        return 4

# Оцінка наявності нерухомості
def evaluate_immovables(has_immovables):
    if has_immovables == 1:
        return 1
    else:
        return 3

# Оцінка наявності транспортного засобу
def evaluate_movables(has_movables):
    if has_movables == 1:
        return 1
    else:
        return 2

# Оцінка типу зайнятості
def evaluate_employment_type(employment_type_id):
    if employment_type_id == 1:
        return 1
    elif 2 <= employment_type_id < 3:
        return 2
    elif 4 <= employment_type_id < 5:
        return 3
    else:
        return 4

# Оцінка щомісячних доходів
def evaluate_monthly_income(monthly_income):
    if monthly_income < 5000:
        return 1
    elif 5000 <= monthly_income < 10000:
        return 2
```



```

elif 10000 <= monthly_income < 20000:
    return 3
elif 20000 <= monthly_income < 40000:
    return 4
else:
    return 5

# Оцінка щомісячних витрат
def evaluate_monthly_expenses(monthly_expenses):
    if monthly_expenses < 5000:
        return 5
    elif 5000 <= monthly_expenses < 10000:
        return 4
    elif 10000 <= monthly_expenses < 20000:
        return 3
    elif 20000 <= monthly_expenses < 40000:
        return 2
    else:
        return 1

# Оцінка gender_id присутніх погашених позик
def evaluate_other_loans_has_closed(other_loans_has_closed):
    if other_loans_has_closed == 1:
        return 2
    else:
        return -2

# Оцінка присутніх непогашених позик
def evaluate_other_loans_active(other_loans_active):
    if other_loans_active == 1:
        return -2
    else:
        return 1

# Оцінка загального розміру непогашених позиків
def evaluate_other_loans_about_current(other_loans_about_current):
    if other_loans_about_current == 0:
        return 3
    elif 1 <= other_loans_about_current < 5000:
        return -1
    elif 5000 <= other_loans_about_current < 10000:
        return -2
    else:
        return -3

```

Кожному ідентифікатору призначатимемо бали в діапазоні від -3 до 5 в залежності від ступеню важливості цього ідентифікатора для прийняття рішення.

Таблиця 1 – Скорингова таблиця

<i><b>Ідентифікатор</b></i>	<i><b>Значення (діапазон)</b></i>	<i><b>Скоринг бал</b></i>
Сума кредиту	Менше 1000	4
	Від 1000 до 5000	3
	Від 5000 до 10000	2
	Більше 10000	1
Термін погашення кредиту	Менше 7 днів	4
	Від 7 до 30 днів	3
	Від 30 до 90 днів	2
	Більше 90 днів	1
Стать	Жінка	2
	Чоловік	1
Кількість дітей	0	1
	Від 1 до 2	2
	Від 3 до 4	3
	Більше 4	4
Рівень освіти	1	1
	Від 2 до 3	2
	Від 4 до 5	3
	Більше 5	4
Наявність нерухомості	Так	3
	Ні	1
Наявність транспорту	Так	2
	Ні	1
Тип зайнятості	1	1
	Від 2 до 3	2
	Від 4 до 5	3
	Більше 5	4

Щомісячні доходи	Менше 5000	1
	Від 5000 до 10000	2
	Від 10000 до 20000	3
	Від 20000 до 40000	4
	Більше 40000	5
Щомісячні витрати	Менше 5000	5
	Від 5000 до 10000	4
	Від 10000 до 20000	3
	Від 20000 до 40000	2
	Більше 40000	1
Присутність погашених позик	Так	2
	Ні	-2
Присутність непогашених позик	Так	-2
	Ні	1
Загальний розмір непогашених позик	0	3
	Від 1 до 5000	-1
	Від 5000 до 10000	-2
	Більше 10000	-3

Зрозуміло, що для реальної задачі такі дані постійно оновлюються з надходженням нового опису позичальника, що не повернув гроші, але в навчальних цілях такого функціоналу буде достатньо.

Додамо вагові коефіцієнти для кожного ідентифікатора та розрахуємо загальний бал як суму всіх балів.

### Лістинг коду:

```
# Функція нарахування балів по кожному параметру
def calculate_score(data):
    # Ваги для кожного показника
    weights = {
        'loan_amount': 0.05,
```

```

        'loan_days': 0.03,
        'gender_id': 0.06,
        'children_count_id': 0.05,
        'education_id': 0.05,
        'has_immovables': 0.04,
        'has_movables': 0.02,
        'employment_type_id': 0.08,
        'monthly_income': 0.07,
        'monthly_expenses': 0.06,
        'other_loans_has_closed': 0.05,
        'other_loans_active': 0.05,
        'other_loans_about_current': 0.04,
    }

    # Обчислення балів для кожного показника
    total_score = 0
    intermediate_scores = {}

    for param, value in data.items():
        if param in weights:
            if param == 'loan_amount':
                param_score = evaluate_loan_amount(value)
            elif param == 'loan_days':
                param_score = evaluate_loan_days(value)
            elif param == 'gender_id':
                param_score = evaluate_gender(value)
            elif param == 'children_count_id':
                param_score = evaluate_children_count(value)
            elif param == 'education_id':
                param_score = evaluate_education(value)
            elif param == 'has_immovables':
                param_score = evaluate_immovables(value)
            elif param == 'has_movables':
                param_score = evaluate_movables(value)
            elif param == 'employment_type_id':
                param_score = evaluate_employment_type(value)
            elif param == 'monthly_income':
                param_score = evaluate_monthly_income(value)
            elif param == 'monthly_expenses':
                param_score = evaluate_monthly_expenses(value)
            elif param == 'other_loans_has_closed':
                param_score = evaluate_other_loans_has_closed(value)
            elif param == 'other_loans_active':
                param_score = evaluate_other_loans_active(value)
            elif param == 'other_loans_about_current':
                param_score = evaluate_other_loans_about_current(value)

            intermediate_scores[param] = param_score * weights[param] # Зберегти
# проміжний бал з урахуванням ваги
            total_score += intermediate_scores[param]

    intermediate_scores['Total_Score'] = total_score
    intermediate_scores['Decision'] = decide_loan_approval(total_score)

    return intermediate_scores

```

Рішення про надання кредиту будемо приймати на основі порогового значення для загального балу. Поставимо такий коефіцієнт 1,2. Але він може бути інший.

### Лістинг коду:

```
# Функція для прийняття рішення щодо видачі кредиту чи відмови
def decide_loan_approval(total_score):
    if total_score >= 1.2:
        return "Approved"
    else:
        return "Denied"
```

Виведемо результати прийняття рішення про надання кредиту.

### Лістинг коду:

```
def credit():
    # Обчислення балів та рішення для кожного клієнта
    results = []
    for index, row in df.iterrows():
        input_data = row.to_dict()
        intermediate_scores = calculate_score(input_data)
        # Зберегти результати
        results.append(intermediate_scores)

    # Виведення та збереження результатів
    result_df = pd.DataFrame(results)
    result_df.to_excel('final_results.xlsx', index=False)
    print(result_df)

    # Розрахунок та відображення відсотків
    approval_percentage = (result_df['Decision'].value_counts()['Approved'] /
len(result_df)) * 100
    denial_percentage = 100 - approval_percentage
    print(f'\nВідсоток клієнтів, яким надали кредит: {approval_percentage:.2f}%')
    print(f'Відсоток клієнтів, яким відмовлено в кредиті: {denial_percentage:.2f}%\n')

    # Відображення scatter графіку
    plt.scatter(result_df['Total_Score'], result_df.index,
                c=result_df['Decision'].apply(lambda x: 1 if x == 'Approved' else 0),
    cmap='viridis')
    plt.title('Scatter графік результатів класифікації')
    plt.xlabel('Total Score')
    plt.ylabel('Клієнт')
    plt.colorbar(ticks=[0, 1], label='Рішення: 0 - Відмова, 1 - Затверджено')
    plt.show()

    return result_df
```

## Результат:

У результаті отримуємо графік розподілу результатів класифікації.

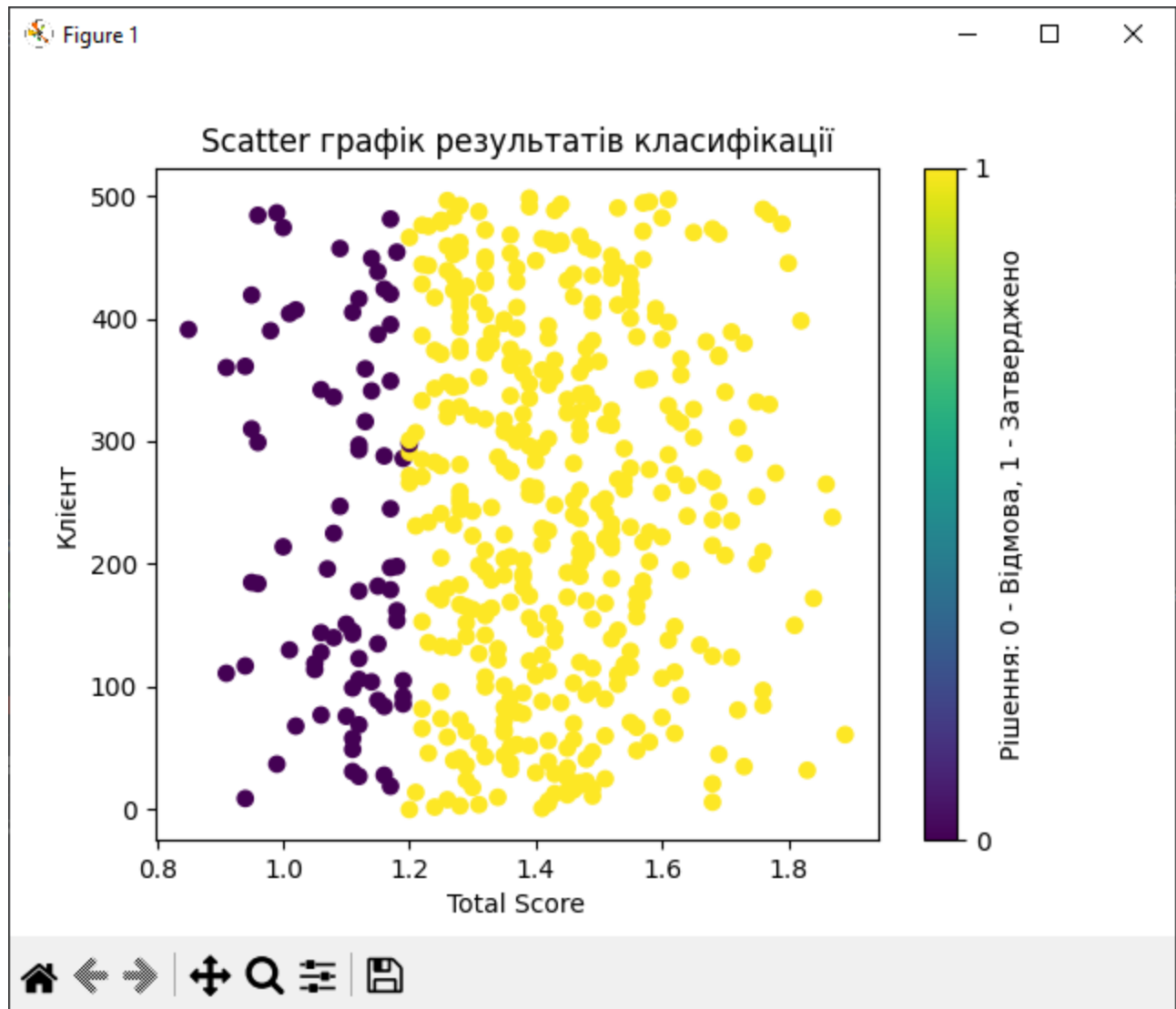


Рисунок 14 – Графік розподілу результату класифікації

Відсоткове співвідношення клієнтів, яким надали кредит та ні.

```
Відсоток клієнтів, яким надали кредит: 83.77%  
Відсоток клієнтів, яким відмовлено в кредиті: 16.23%
```

Рисунок 15 – Відсоток клієнтів, яким було відмовлено

А також таблицю зі скорінговими балами та результатом класифікації для кожного клієнта.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	loan_amount	loan_days	gender_id	children_count_id	education_id	has_immovables	has_movables	employment_type_id	monthly_income	monthly_expenses	other_loans_has_closed	other_loans_active	other_loans_about_current	Total_Score	Decision
2	0,15	0,06	0,12	0,1	0,2	0,04	0,02	0,08	0,21	0,3	-0,1	-0,1	0,12	1,2	Approved
3	0,15	0,09	0,12	0,2	0,15	0,04	0,02	0,08	0,21	0,24	0,1	0,05	-0,04	1,41	Approved
4	0,15	0,12	0,06	0,1	0,15	0,04	0,02	0,08	0,21	0,24	-0,1	0,05	0,12	1,24	Approved
5	0,15	0,06	0,12	0,1	0,2	0,04	0,02	0,08	0,14	0,3	-0,1	0,05	0,12	1,28	Approved
6	0,15	0,09	0,06	0,1	0,2	0,12	0,02	0,08	0,14	0,24	0,1	0,05	-0,04	1,31	Approved
7	0,15	0,06	0,06	0,1	0,2	0,04	0,02	0,08	0,14	0,3	0,1	0,05	0,12	1,42	Approved
8	0,15	0,06	0,12	0,1	0,2	0,12	0,02	0,32	0,28	0,24	-0,1	0,05	0,12	1,68	Approved
9	0,15	0,09	0,12	0,2	0,2	0,04	0,02	0,08	0,21	0,24	-0,1	0,05	0,12	1,42	Approved
10	0,15	0,09	0,12	0,1	0,2	0,04	0,02	0,08	0,21	0,18	-0,1	0,05	0,12	1,26	Approved
11	0,15	0,09	0,06	0,1	0,2	0,04	0,02	0,08	0,14	0,3	-0,1	-0,1	-0,04	0,94	Denied
12	0,15	0,06	0,06	0,15	0,2	0,04	0,02	0,32	0,21	0,3	-0,1	0,05	-0,12	1,34	Approved
13	0,15	0,12	0,06	0,1	0,15	0,04	0,02	0,32	0,28	0,18	-0,1	0,05	0,12	1,49	Approved
14	0,15	0,06	0,12	0,2	0,2	0,04	0,02	0,08	0,21	0,3	-0,1	0,05	0,12	1,45	Approved
15	0,15	0,06	0,12	0,1	0,15	0,04	0,02	0,08	0,14	0,3	0,1	0,05	0,12	1,43	Approved
16	0,15	0,06	0,12	0,1	0,1	0,04	0,02	0,24	0,21	0,3	-0,1	0,05	-0,08	1,21	Approved
17	0,15	0,06	0,12	0,1	0,2	0,04	0,02	0,32	0,07	0,3	-0,1	0,05	0,12	1,45	Approved
18	0,2	0,12	0,12	0,1	0,2	0,12	0,02	0,08	0,14	0,24	0,1	-0,1	0,12	1,46	Approved
19	0,15	0,09	0,06	0,1	0,2	0,04	0,04	0,08	0,28	0,18	0,1	0,05	0,12	1,49	Approved
20	0,15	0,06	0,12	0,1	0,15	0,04	0,02	0,08	0,21	0,3	-0,1	0,05	0,12	1,3	Approved
21	0,15	0,06	0,12	0,1	0,2	0,04	0,02	0,08	0,14	0,3	0,1	-0,1	-0,04	1,17	Denied
22	0,15	0,09	0,12	0,1	0,2	0,04	0,02	0,24	0,14	0,3	-0,1	0,05	0,12	1,47	Approved
23	0,15	0,09	0,12	0,2	0,1	0,12	0,02	0,32	0,21	0,24	0,1	0,05	-0,04	1,68	Approved
24	0,15	0,06	0,12	0,2	0,2	0,04	0,02	0,16	0,21	0,24	-0,1	0,05	0,12	1,47	Approved
25	0,15	0,09	0,12	0,2	0,2	0,04	0,02	0,08	0,21	0,3	-0,1	0,05	0,12	1,48	Approved
26	0,15	0,06	0,12	0,1	0,2	0,04	0,02	0,08	0,21	0,24	-0,1	0,05	0,12	1,29	Approved
27	0,15	0,09	0,06	0,1	0,15	0,04	0,02	0,32	0,21	0,3	-0,1	0,05	0,12	1,51	Approved
28	0,15	0,09	0,12	0,1	0,1	0,04	0,02	0,32	0,14	0,3	-0,1	0,05	0,12	1,45	Approved
29	0,2	0,06	0,06	0,2	0,2	0,04	0,02	0,08	0,07	0,12	-0,1	0,05	0,12	1,12	Denied
30	0,15	0,09	0,06	0,1	0,2	0,04	0,02	0,08	0,28	0,18	0,1	-0,1	-0,04	1,16	Denied

Рисунок 16 – Результуюча таблиця зі скорінговим балом

## 2. 4. Детекція шахрайства

Виконаємо пошук аномалій у вибірці за скорінговим балом та виведемо результати в графіках.

### Лістинг коду:

```
# Функція для виявлення шахрайства
def fraud_detection(X):
    outliers_fraction = 0.1
    clf = KNN(contamination=outliers_fraction)
    clf.fit(X)
    y_pred = clf.predict(X)
    return y_pred

# Функція для відображення графіка шахрайства
def plot_fraud_detection(X, y_pred):
    norm_data = StandardScaler().fit_transform(X)
    compressed = PCA(n_components=2).fit_transform(norm_data)
    plt.figure(figsize=(10, 5))
    sns.scatterplot(x=compressed[:, 0], y=compressed[:, 1], hue=np.where(y_pred, "fraud",
"no fraud"))
    plt.show()

# Функція для підрахунку відсотків шахрайства
def calculate_fraud_percentages(y_pred):
    fraud_percentage = (y_pred.sum() / len(y_pred)) * 100
```

```

no_fraud_percentage = 100 - fraud_percentage
return fraud_percentage, no_fraud_percentage

def fraud(result_df):
    X = result_df.drop(['Total_Score', 'Decision'], axis=1) # Виключаємо колонки з
результатами
    y_pred_fraud = fraud_detection(X)

    # Додаємо колонку з передсказаною міткою шахрайства в результати
    result_df['Fraud_Prediction'] = y_pred_fraud

    # Вивід результатів
    print(result_df)

    # Підрахунок та відображення відсотків шахрайства
    fraud_percentage, no_fraud_percentage = calculate_fraud_percentages(y_pred_fraud)
    print(f'\nВідсоток клієнтів, які виявлені як шахраї: {fraud_percentage:.2f}%')
    print(f'Відсоток клієнтів, які не виявлені як шахраї: {no_fraud_percentage:.2f}%\n')

    # Відображення scatter графіку для виявлення шахрайства
    plot_fraud_detection(X, y_pred_fraud)

    return

```

## Результат:

У результаті отримуємо наступні показники для шахрайства. Графік розподілу шахраїв серед всіх клієнтів.

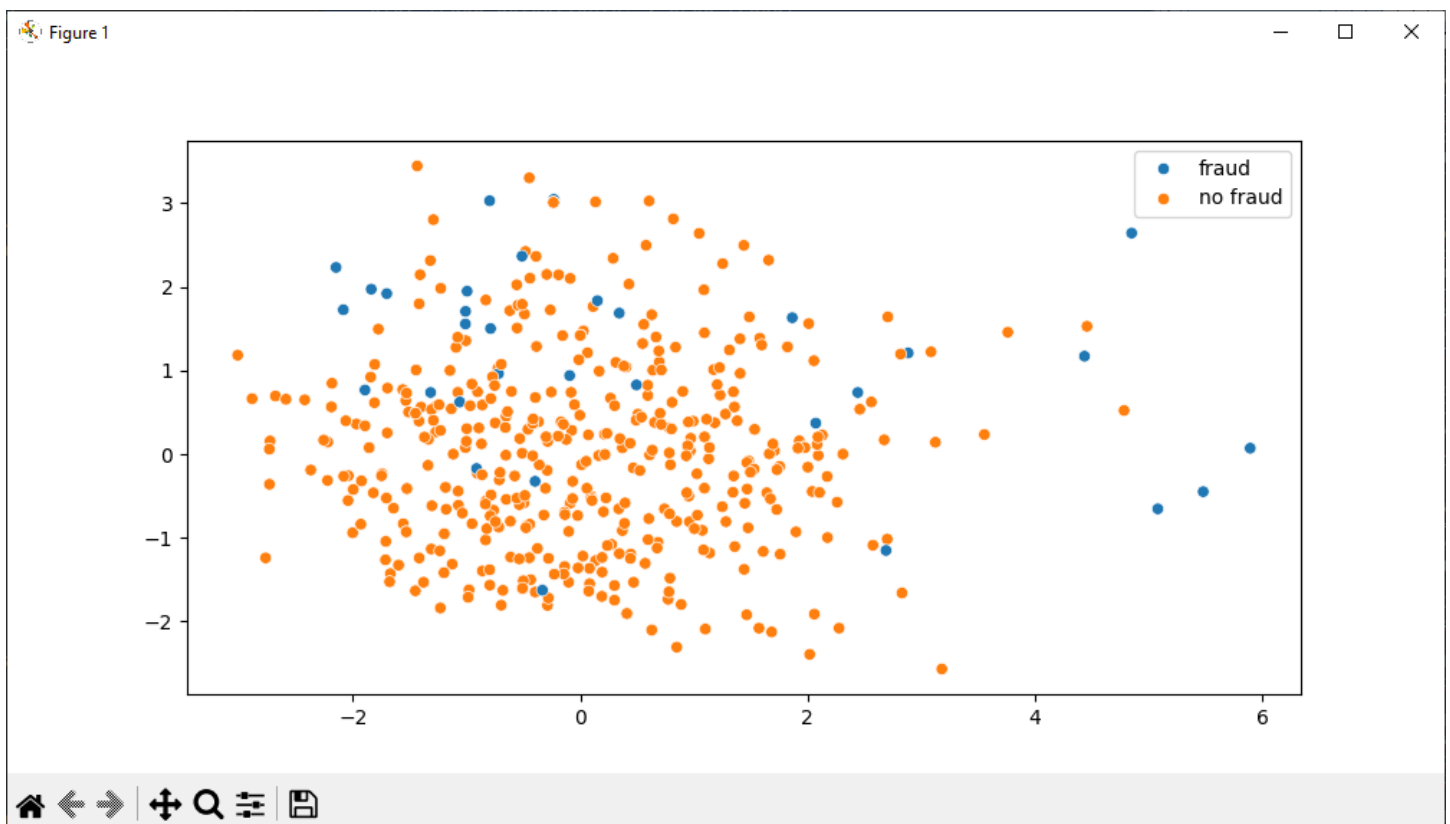


Рисунок 17 – Результат детекції шахрайства



Відсоткове співвідношення шахраїв.

```
Відсоток клієнтів, які виявлені як шахраї: 6.61%  
Відсоток клієнтів, які не виявлені як шахраї: 93.39%
```

Рисунок 18 – Відсоток шахраїв

Результати по шахрайству для конкретних клієнтів.

```
   loan_amount  loan_days  gender_id  ...  Total_Score  Decision  Fraud_Prediction  
0           0.15       0.06       0.12  ...           1.20  Approved           1  
1           0.15       0.09       0.12  ...           1.41  Approved           0  
2           0.15       0.12       0.06  ...           1.24  Approved           0  
3           0.15       0.06       0.12  ...           1.28  Approved           0  
4           0.15       0.09       0.06  ...           1.31  Approved           0  
..          ...        ...        ...  ...          ...        ...          ...  
494         0.15       0.09       0.12  ...           1.57  Approved           0  
495         0.20       0.12       0.06  ...           1.58  Approved           0  
496         0.15       0.06       0.06  ...           1.26  Approved           0  
497         0.15       0.09       0.12  ...           1.61  Approved           0  
498         0.15       0.12       0.12  ...           1.39  Approved           0  
[499 rows x 16 columns]
```

Рисунок 19 – Результати шахрайства для кожного клієнта

## 2.5. Аналіз отриманих результатів

Розроблено скорингову карту для прийняття рішення щодо надання кредиту клієнту. Підтвердження – таблиця 1.

Розроблено програмний скрипт для первинної обробки даних (скорингова карта, ідентифікатори скорингової карти). Підтвердження – рисунки 20-23.

#	AU	AV	AW	AX	AY	AZ	BA	BB	BC	BD	BE	BF	BG	BH	BI	BJ	BK	BL	BM	BN	BO
1	amount_to_pay	total_paid	current_interest	current_overdue	current_rest_amount	current_foverdue	overdue	extensor	extensor	payment_date	closed_at		product	credit	pouser_id	face_id	prolongat	prolongat	wizard	tystep	created_at
2	0	4620	1620	0	0	0	0	0	0	0	02.03.2021	01.03.2021 16:17	14	31	253430	NULL	NULL	NULL	2	3	#####
3	0	1120	120	0	0	0	0	0	0	0	07.02.2021	07.02.2021 15:30	12	38	109618	NULL	NULL	NULL	2	3	#####
4	1088	1183	145	38	0	0	0	2	0	0	03.02.2021	05.02.2021 18:26	14	39	289130	NULL	NULL	NULL	5	7	#####
5	0	2168,4	568,4	0	0	0	0	0	0	0	02.03.2021	18.02.2021 18:42	18	38	141625	226701	NULL	NULL	2	3	#####
6	3402,5	11290	5322,5	3467,5	0	0	0	73	0	0	19.04.2021	23.05.2021 18:29	14	39	204349	NULL	1	20	5	7	#####
7	0	1472,48	443,8	28,68	0	0	0	2	0	0	03.04.2021	11.03.2021 15:22	14	31	122083	NULL	1	30	5	7	#####
8	2028,6	3435,2	1407,2	728	0	0	0	28	0	0	02.03.2021	30.03.2021 10:24	18	38	225205	NULL	NULL	NULL	2	3	#####
9	0	2506	506	0	0	0	0	0	0	0	15.02.2021	13.02.2021 11:02	14	39	209131	NULL	NULL	NULL	5	7	#####
10	0	2847	347	0	0	0	0	0	0	0	12.02.2021	08.02.2021 21:13	14	31	276402	NULL	NULL	NULL	2	3	#####
11	0	1120	120	0	0	0	0	0	0	0	07.02.2021	03.02.2021 9:55	18	38	1777	NULL	NULL	NULL	2	3	#####
12	0	2476	476	0	0	0	0	0	0	0	03.03.2021	12.02.2021 4:14	14	31	223487	NULL	NULL	NULL	2	3	#####
13	0	2161,2	705,2	56	0	0	0	2	0	0	07.03.2021	26.02.2021 23:17	14	31	250637	NULL	1	30	2	3	#####
14	3927,5	8630	3802,5	2327,5	0	0	0	49	0	0	03.03.2021	21.04.2021 15:52	14	39	289142	NULL	NULL	NULL	5	7	#####
15	0	2879,4	879,4	0	0	0	0	0	0	0	03.03.2021	02.03.2021 10:39	14	31	116802	NULL	NULL	NULL	2	3	#####
16	2215,55	2406,5	807,35	99,15	0	0	0	3	0	0	03.03.2021	06.03.2021 4:25	14	31	241322	NULL	NULL	NULL	2	3	#####
17	0	1318	318	0	0	0	0	0	0	0	03.03.2021	17.02.2021 5:42	14	31	24870	NULL	NULL	NULL	2	3	#####
18	0	876,8	76,8	0	0	0	0	0	0	0	04.02.2021	NULL	14	31	236645	NULL	NULL	NULL	2	3	#####
19	1422,8	1528,6	287,6	41	0	0	0	2	0	0	10.02.2021	12.02.2021 16:11	14	31	281395	NULL	NULL	NULL	2	3	#####
20	4620	6156	2376	780	0	0	0	13	0	0	03.03.2021	16.03.2021 13:19	14	31	235734	NULL	NULL	NULL	2	3	#####
21	0	2245,7	295,7	0	0	0	0	0	0	0	03.03.2021	08.02.2021 4:00	14	31	106816	NULL	NULL	NULL	2	3	#####
22	0	1867,4	167,4	0	0	0	0	0	0	0	10.02.2021	05.02.2021 2:20	14	31	86072	194019	NULL	NULL	2	3	#####
23	0	4145	1095	0	0	0	0	0	0	0	21.02.2021	21.02.2021 18:06	14	31	153435	237604	NULL	NULL	2	3	#####
24	0	3459,5	1874	85,5	0	0	0	3	0	0	05.05.2021	06.04.2021 21:09	14	39	289167	NULL	2	60	5	7	#####
25	1806,5	2006	420,5	85,5	0	0	0	3	0	0	11.02.2021	14.02.2021 14:46	14	39	286897	NULL	NULL	NULL	5	7	#####
26	0	1464,7	464,7	0	0	0	0	0	0	0	03.03.2021	03.03.2021 13:21	14	39	284922	NULL	NULL	NULL	5	7	#####
27	0	3648,5	648,5	0	0	0	0	0	0	0	17.02.2021	17.02.2021 23:13	14	31	263804	NULL	NULL	NULL	2	3	#####
28	1739	2333	536	297	0	0	0	10	0	0	09.03.2021	19.02.2021 20:53	14	31	272420	NULL	NULL	NULL	2	3	#####
29	0	568	68	0	0	0	0	0	0	0	03.03.2021	01.02.2021 23:29	14	31	254736	NULL	NULL	NULL	2	3	#####
30	0	2126	126	0	0	0	0	0	0	0	15.02.2021	02.02.2021 13:01	14	39	289188	NULL	NULL	NULL	5	7	#####

Рисунок 20 – Необроблена скорингова карта

#	B	C	D	E	F
1	Description of information	Filling	At the time of the request	Place of definition	Note
2	Ідентифікатор заявки	Обов'язково	Відомо	Визначається системою обліку заявок на момент реєстрації	
3	Сума кредиту в заявці	Обов'язково	Відомо	Вказує позичальник	
4	Термін погашення кредиту в заявці	Обов'язково	Відомо	Вказує позичальник	
5	Дата і час прийняття заявки	Обов'язково	Відомо	Визначається при подачі заявки	Локальний час на пристрої користувача або час прийому заявок?
6				Визначається браузером при подачі заявки	Завжди UA
7	ID мети використання кредиту	Обов'язково	Відомо	Вказує позичальник	Завжди 999 – 999 це клієнт вписує саме призначення, раніше можна було вибирати зі списку
8	Мета використання кредиту	Обов'язково	Відомо	Вказує позичальник	
9	IP клієнта, з якого надійшов запит	Обов'язково	Відомо	Визначається браузером при подачі заявки	
10	Технічна інформація про пристрій клієнта	Обов'язково	Відомо	Визначається браузером при подачі заявки	
11	Сайт з якого користувач прийшов на сторінку заявки	Відомо	Відомо	Визначається браузером при подачі заявки	
12	не використовується				Завжди пусто
13	Стать позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
14	Дата народження позичальника	Обов'язково	Відомо	Вказує позичальник	Іноді заповнюється не вірно
15	Сімейний статус позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
16	Кількість дітей у позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
17	Рівень освіти позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
18	Номер паспорта	Обов'язково	Відомо	Вказує позичальник	Повинен бути заповнений
19	Дата видачі паспорта	Обов'язково	Відомо	Вказує позичальник	Повинна бути заповнена вірно
20	Кім виданий паспорт		Відомо	Вказує позичальник	
21	ИНН позичальника	Обов'язково	Відомо	Вказує позичальник	Повинен бути заповнений
22	email позичальника	Обов'язково	Відомо	Вказує позичальник	Повинен бути заповнений
23	Мобільний телефон позичальника	Обов'язково	Відомо	Вказує позичальник	Повинен бути заповнений
24	Посилання на профілі в соціальних мережах		Відомо	???	Будет ли информация на момент заявки?
25	Адреса реєстрації: область	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
26	Адреса реєстрації: поштовий індекс		Відомо	Вказує позичальник	
27	Адреса реєстрації: місто		Відомо	Вказує позичальник	
28	Адреса реєстрації: тип вулиці		Відомо	Вказує позичальник	
29	Адреса реєстрації: вулиця		Відомо	Вказує позичальник	
30	Адреса реєстрації: номер дома		Відомо	Вказує позичальник	

Рисунок 21 – Необроблені ідентифікатори скорингової карти

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1		loan_amount	loan_days	gender_id	birth_date	children_count	education_id	owner_type	immovables	movables	employment_type	organization_type	organization_branch	employees_count	seniority_years	monthly_income	income_frequency	monthly_expenses	other_loans_has_closed	other_loans_active	other_loans_about_current	other_loans_about_monthly	product_id
2	0	3000	30	1	#####	1	5	4	0	0	1	1	0	0	5	15000	2	1	4000	0	1	0	4263
3	1	1000	7	1	#####	2	4	0	0	0	1	4	0	0	8	11000	2	1	5000	1	2	1500	800
4	2	1000	3	2	#####	1	4	2	0	0	1	999	0	0	5	10000	2	1	5000	0	0	0	0
5	3	1600	30	1	#####	1	3	3	0	0	1	1	0	0	3	8000	2	1	3000	0	0	0	0
6	4	2500	18	2	#####	1	5	3	1	0	1	2	0	0	1	9000	2	1	8000	1	2	2000	1500
7	5	1000	30	2	#####	1	3	3	0	0	1	2	0	0	1	6500	2	1	3000	1	0	0	0
8	6	1300	30	1	#####	1	5	1	1	0	3	0	0	0	5	20000	999	2	7000	0	0	0	0
9	7	2000	14	1	#####	2	5	3	0	0	1	2	0	0	13	14000	2	1	7000	0	0	0	0
10	8	2500	12	1	#####	1	5	1	0	0	1	999	0	0	10	15000	1	1	12000	0	0	0	0
11	9	1000	7	2	#####	1	6	4	0	0	1	999	8	2	4	7200	2	1	4000	0	1	300	150
12	10	2000	30	2	#####	3	3	1	0	0	3	999	0	0	14	15000	2	1	2000	0	3	20000	15000
13	11	1400	3	2	#####	1	4	2	0	0	8	0	999	1	6	25000	2	999	15000	0	0	0	0
14	12	2500	30	1	#####	2	3	2	0	0	1	4	0	0	8	14000	2	1	4500	0	0	0	0
15	13	2000	30	1	#####	1	4	2	0	0	1	3	0	0	1	9000	1	1	3000	1	0	0	0
16	14	1500	30	1	#####	1	2	1	0	0	4	0	0	0	1	10000	1	999	3000	0	3	5000	500
17	15	1000	30	1	#####	1	5	3	0	0	3	1	0	0	1	2000	2	1	500	0	0	0	0
18	16	800	3	1	#####	1	5	1	1	0	1	999	0	0	20	9000	2	1	5000	1	1	0	500
19	17	1200	9	2	#####	1	3	2	0	1	1	999	0	0	8	30000	2	999	15999	1	0	0	0
20	18	3000	30	1	#####	1	4	3	0	0	1	999	0	0	8	10000	1	1	3000	0	0	0	0
21	19	1950	30	1	#####	1	3	4	0	0	1	999	0	0	2	7900	1	1	4500	1	1	2000	1000
22	20	1700	9	1	#####	1	5	1	0	0	4	0	0	0	5	5000	2	1	3500	0	0	0	0
23	21	3050	20	1	#####	2	2	2	1	0	3	999	0	0	15	12000	2	1	5000	1	3	3000	3000
24	22	1500	30	1	#####	2	3	1	0	0	2	4	0	0	15	10000	1	999	5000	0	0	0	0
25	23	1500	10	1	#####	2	5	3	0	0	1	999	0	0	13	12000	2	1	2000	0	0	0	0
26	24	1000	30	1	#####	1	5	1	0	0	1	999	0	0	1	10000	1	1	5000	0	0	0	0
27	25	3000	16	2	#####	1	4	3	0	0	3	999	0	0	4	10000	2	1	3000	0	0	0	0
28	26	1500	8	1	#####	1	2	3	0	0	6	0	0	0	1	5600	2	1	3400	0	0	0	0
29	27	500	30	2	#####	2	3	2	0	0	1	2	0	0	3	3100	2	1	31000	0	0	0	0
30	28	2000	14	2	#####	1	5	4	0	0	1	2	0	0	3	20000	2	1	10000	1	1	4000	300
31	29	1000	7	1	#####	1	5	1	1	0	1	999	0	0	7	8000	2	2	3000	1	2	2000	2000
32	30	1000	10	1	#####	1	5	3	0	0	2	999	0	0	12	12000	2	4	8000	0	0	0	2000

Рисунок 22 – Оброблена скриньова карта

	A	B	C	D	E	F	G
1		Field_in_data	Description_of_information	Filling	At the time of the request	Place_of_definition	Note
2	0	loan_amount	Сума кредиту в заявці	Обов'язково	Відомо	Вказує позичальник	
3	1	loan_days	Термін погашення кредиту в заявці	Обов'язково	Відомо	Вказує позичальник	
4	2	gender_id	Стать позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
5	3	birth_date	Дата народження позичальника	Обов'язково	Відомо	Вказує позичальник	Іноді заповнюється не вірно
6	4	children_count_id	Кількість дітей у позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
7	5	education_id	Рівень освіти позичальника	Обов'язково	Відомо	Вказує позичальник	Потрібна расшифровка id
8	6	fact_addr_owner_type_id	Адрес проживання: тип власності		Відомо	Вказує позичальник	Потрібна расшифровка id
9	7	has_immovables	Володіє нерухомістю	Обов'язково	Відомо	Вказує позичальник	
10	8	has_movables	Володіє транспортним засобом	Обов'язково	Відомо	Вказує позичальник	
11	9	employment_type_id	Місце роботи: Тип зайнятості		Відомо	Вказує позичальник	Потрібна расшифровка id
12	10	organization_type_id	Місце роботи: тип організації		Відомо	Вказує позичальник	Потрібна расшифровка id
13	11	organization_branch_id	Місце роботи: галузь організації		Відомо	Вказує позичальник	Потрібна расшифровка id
14	12	employees_count_id	Місце роботи: кількість співробітників організації		Відомо	Вказує позичальник	Потрібна расшифровка id
15	13	seniority_years	Трудовий стаж	Обов'язково	Відомо	Вказує позичальник	Повинен бути вказаний вірно
16	14	monthly_income	Щомісячні доходи	Обов'язково	Відомо	Вказує позичальник	
17	15	income_frequency_id	Частота отримання доходів	Обов'язково	Відомо	Вказує позичальник	
18	16	income_source_id	Джерело отримання доходів	Обов'язково	Відомо	Вказує позичальник	
19	17	monthly_expenses	Щомісячні витрати	Обов'язково	Відомо	Вказує позичальник	
20	18	other_loans_has_closed	Усі попередні позики погашені		Відомо	Вказує позичальник	
21	19	other_loans_active	Присутні непогашені позики		Відомо	Вказує позичальник	
22	20	other_loans_about_current	Загальний розмір непогашених позиків		Відомо	Вказує позичальник	
23	21	other_loans_about_monthly	Щомісячні виплати по непогашеним позикам		Відомо	Вказує позичальник	
24	22	product_id	id продукту, який видається замовнику	Обов'язково	???	параметри, пов'язані з виданим продуктом	
25	23	product_dpr	відсоткова ставка по кредиту		???	параметри, пов'язані з виданим продуктом	
26	24	product_amount_from	Нижня межа виду позики	Обов'язково	???	параметри, пов'язані з виданим продуктом	
27	25	product_amount_to	Верхня межа виду позики	Обов'язково	???	параметри, пов'язані з виданим продуктом	
28	26	product_interest_min	мінімальна сума по продукту		???	параметри, пов'язані з виданим продуктом	
29	27	product_base_amount_limit	ліміт суми по продукту		???	параметри, пов'язані з виданим продуктом	
30	28	amount_limit	ліміт суми		???	параметри, пов'язані з виданим продуктом	
31	29	applied_limit	затверджені ліміти		???	параметри, пов'язані з виданим продуктом	
32	30	amount_to_pay	сума для оплати мінімальна		???	параметри, пов'язані з виданим продуктом	

Рисунок 23 – Оброблені ідентифікатори скорингової карти

Розроблено програмний скрипт для реалізації скорингового аналізу на основі евристичних фнккції. Підтвердження – прийняті рішення про надання або не надання кредитів.

```
Відсоток клієнтів, яким надали кредит: 83.77%  
Відсоток клієнтів, яким відмовлено в кредиті: 16.23%
```

Рисунок 24 – Відсоткові результати надання кредитів клієнтам

Розроблено програмний скрипт для детекції шахрайства з використанням виявлення аномалій. Підтвердження – прийняті рішення про виявлення шахрайства.

```
Відсоток клієнтів, які виявлені як шахраї: 6.61%  
Відсоток клієнтів, які не виявлені як шахраї: 93.39%
```

Рисунок 25 – Відсоткові результати виявлених шахраїв

Використано інший підхід до реалізації поставленої задачі.

### **Висновок:**

Під час виконання лабораторної роботи досліджено процес скорингового банківського аналізу та скорингу шахрайства. На основі отриманих даних сформульовано задачу для обох областей з урахуванням отриманих вхідних даних.

Розроблено програмний скрипт, що реалізовує первинну обробку даних та скоринговий аналіз даних від позичальників та ідентифікаторів скорингової карти. Додатково сформульовано приклад скорингової карти, за допомогою якої було виконано нарахування балів для кожного ідентифікатора та прийнято рішення про надання кредиту чи відмові. Для виконання задачі скорингового аналізу даних використано інший підхід – застосування евристичних функцій для оцінки скорингового балу індикаторів.

Розроблено програмний скрипт, що реалізовує детекцію шахрайства на основі виявлення аномальних значень у вибірці.

Обидва скрипти протестовані на вхідних даних. У результаті отримано цілком реалістичні результати. Отже можна зробити висновок, що завдання виконано успішно.