

RFM-анализ клиентской базы с использованием Python

Мандронов Антон Андреевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В данном исследовании рассматривается применение RFM-анализа с использованием языка Python для сегментации клиентской базы. С помощью нейросети DeepSeek был сгенерирован синтетический датасет, соответствующий заданным параметрам: 500 клиентов, 1000 товаров, 100 000 транзакций с диапазоном стоимостей 100–5000 рублей за 2024 год. Разработан алгоритм автоматического расчёта RFM-метрик (Recency, Frequency, Monetary) с классификацией клиентов по трёхбалльной шкале и распределением по сегментам. Реализация кода в Google Colab позволила автоматизировать обработку данных и визуализацию результатов, включая распределение клиентов и тепловую карту метрик. Исследование подтверждает эффективность Python для RFM-анализа, что делает его ценным инструментом для маркетинговой аналитики.

Ключевые слова: RFM-анализ, Python, сегментация клиентов, обработка данных, машинное обучение, маркетинговая аналитика, Google Colab, визуализация данных, DeepSeek, синтетические данные.

RFM analysis of customer base using python

Mandronov Anton Andreevich

Sholom-Aleichem Priamursky State University

Student

Abstract

This study examines the application of RFM analysis using the Python programming language for customer base segmentation. A synthetic dataset was generated using the DeepSeek neural network, meeting specified parameters: 500 customers, 1,000 products, and 100,000 transactions with a price range of 100–5,000 rubles for the year 2024. An algorithm was developed to automatically calculate RFM metrics (Recency, Frequency, Monetary), classifying customers on a three-point scale and segmenting them accordingly. The implementation of the code in Google Colab enabled automated data processing and visualization of results, including customer distribution and a heatmap of metrics. The study confirms the effectiveness of Python for RFM analysis, making it a valuable tool for marketing analytics.

Keywords: RFM analysis, Python, customer segmentation, data processing, machine learning, marketing analytics, Google Colab, data visualization, DeepSeek, synthetic data

1 Введение

1.1 Актуальность

В современных условиях растущей конкуренции и увеличения затрат на привлечение новых клиентов особую важность приобретают методы анализа и сегментации существующей клиентской базы. RFM-анализ, оценивающий клиентов по трем ключевым параметрам - давности (Recency), частоте (Frequency) и сумме покупок (Monetary), остается одним из наиболее эффективных инструментов для выявления ценных групп потребителей. Применение Python для реализации RFM-анализа особенно актуально благодаря его способности обрабатывать большие массивы данных, гибкости в настройке параметров анализа, возможности интеграции с методами машинного обучения, а также автоматизации процессов визуализации и отчетности. Это делает Python мощным инструментом для маркетологов и аналитиков, стремящихся оптимизировать клиентские стратегии.

1.2 Обзор исследований

В работе Е. П. Голубкова рассматривается методика RFM-анализа и её применение в маркетинге, включая ключевые этапы сегментации клиентов. Особое внимание уделено интерпретации результатов для повышения эффективности маркетинговых кампаний [1]. В исследовании А. В. Ткаченко показано, как RFM-анализ помогает увеличить повторные продажи за счёт персонализации взаимодействия с клиентами. Приводятся примеры успешного внедрения методики в интернет-маркетинге [2]. В публикации А.Н. Михайлова анализируется роль Python в обработке CRM-данных для оптимизации клиентских стратегий, включая автоматизацию RFM-анализа. Подчеркивается важность программирования для точного прогнозирования поведения покупателей [3]. В работе В. И. Александрова исследуется применение RFM-анализа в e-commerce для разработки таргетированных маркетинговых стратегий. Рассматриваются кейсы, подтверждающие его эффективность в повышении конверсии и лояльности клиентов [4].

1.3 Цель исследования

Целью исследования является разработка и практическое применение методики RFM-анализа с использованием языка программирования Python.

2 Материалы и методы

Для исследования была применена современная модель искусственного интеллекта DeepSeek, специализирующаяся на генерации реалистичных бизнес-данных. Синтетический датасет представлен в виде структурированной таблицы, содержащей следующие ключевые поля:

- Идентификатор клиента
- Дата покупки
- Стоимость покупки

Требования к данным:

- 500 клиентов
- 1000 товаров
- 100000 транзакций
- Диапазон стоимостей: 100-5000 рублей
- Даты покупки за 2024 год

Для отладки кода на Python был использован бесплатный облачный сервис от Google - Google Colab. Версия Python, используемая в Colab - 3.11.12.

3 Результаты и обсуждения

В рамках исследования был сформирован запрос к нейросети DeepSeek: *«Сгенерируй данные для RFM-анализа содержащий три столбца: Id клиента, дата покупки и стоимость покупки. Необходимые требования к данным: 500 клиентов, 1000 товаров, 100000 транзакций, стоимость покупки варьируется от 100 по 5000, покупки совершались за 2024 год»*

На основании данного запроса был сформирован Python код, использующий библиотеки Pandas для создания датасетов и формирования Excel таблиц, NumPy и Datetime для генерации значений. Для запуска кода воспользуемся Google Colab (рис. 1), который позволяет писать и запускать Python-код прямо в браузере без необходимости устанавливать что-либо на компьютер.

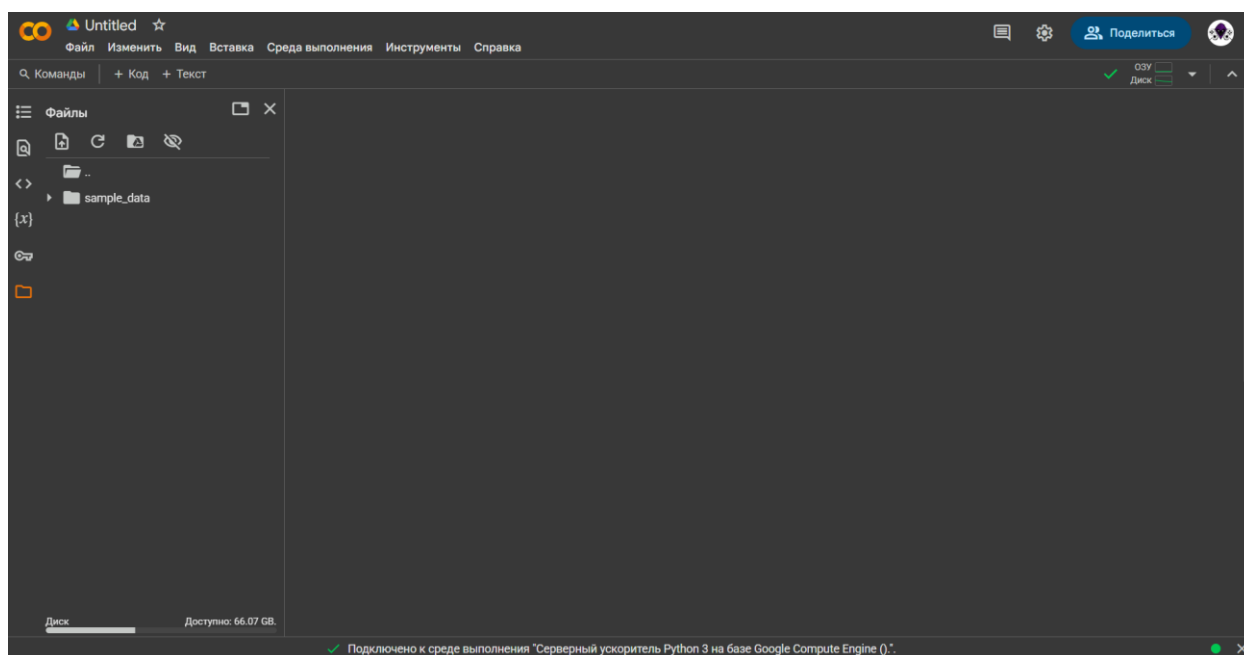


Рис 1. Google Colab

Создаём блок для кода через кнопку «+ Код» и вставляем код, сгенерированный нейросетью. Для запуска нажимаем сочетание клавиш Ctrl+Enter. В разделе «Файлы» появляется сгенерированный файл rfm_data_2024.xlsx (рис. 2).

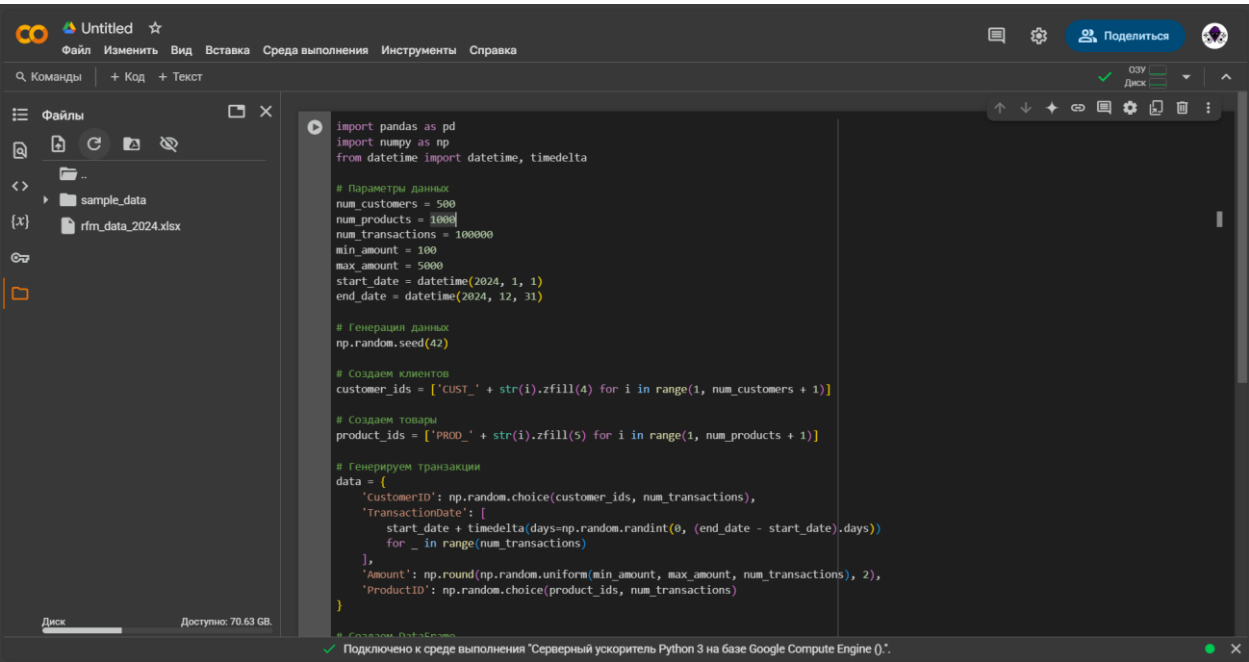


Рис 2. Блок с кодом и сгенерированной таблицей

Дважды нажимаем на rfm_data_2024.xlsx для скачивания файла на компьютер.
Открываем файл в Excel (рис. 3).

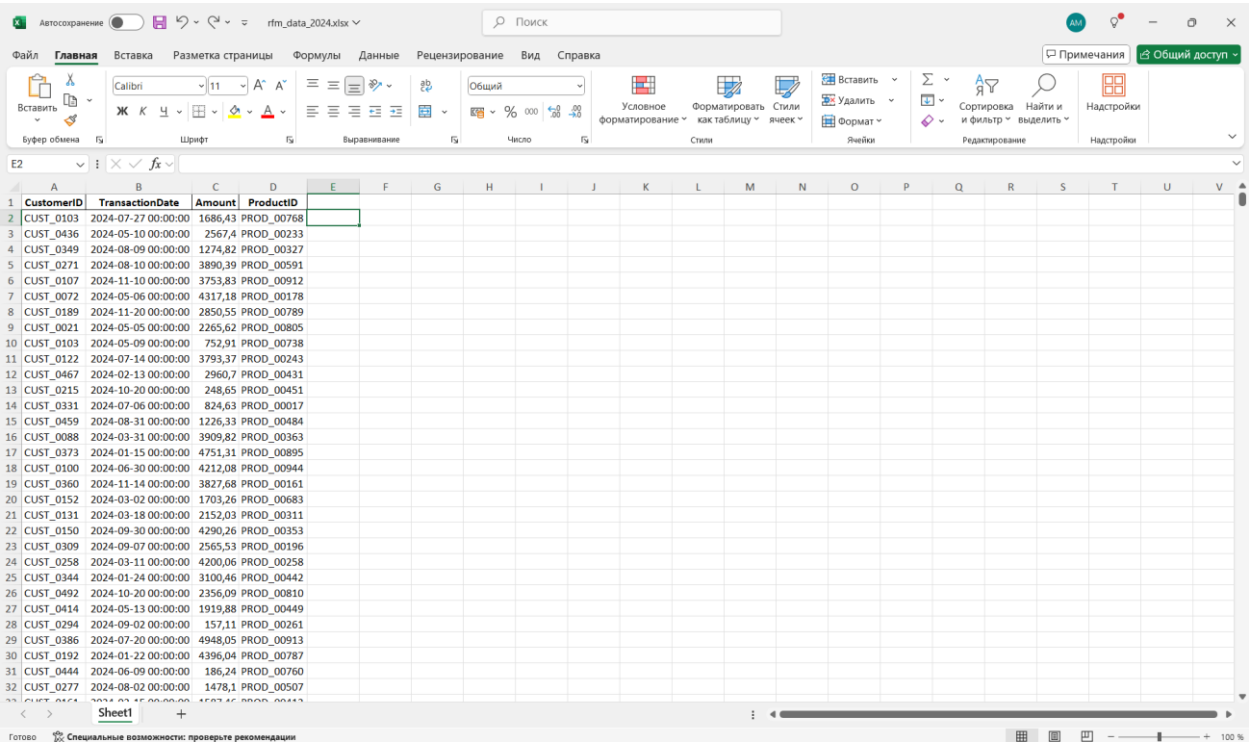


Рис 3. rfm_data_2024.xlsx

Перед началом анализа проверим, что данные соответствуют запросу:

- **Количество клиентов:** Чтобы определить количество клиентов в сгенерированных данных, необходимо подсчитать количество уникальных

значений в столбце клиентов. Для этого в ячейку E2 вставим формулу «=СЧЁТЗ(УНИК(A2:A100001))». Число клиентов – 500 (рис. 4).

	C	D	E	F
	Amount	ProductID	Количество клиентов	
0	1686,43	PROD_00768	500	
0	2567,4	PROD_00233		
0	1274,82	PROD_00327		

Рис 4. Количество клиентов

• **Количество товаров:** Как и в случае с клиентами, считаем количество уникальных в ячейке F2 по формуле «=СЧЁТЗ(УНИК(D2:D100001))» (рис. 5).

	D	E	F	G
	ProductID	Количество клиентов	Количество товаров	
3	PROD_00768	500	1000	
4	PROD_00233			
5	PROD_00327			

Рис 5. Количество товаров

• **Количество транзакций:** В ячейке G2 вставляем «=СЧЁТЗ(A2:A100001))» и получаем число транзакций в таблице (рис. 6).

E	F	G	H
Количество клиентов	Количество товаров	Количество транзакций	
500	1000	100000	

Рис 6. Количество транзакций

• **Диапазон стоимостей:** Найдём минимальное и максимальное значения в столбце Amount, чтобы убедиться, что диапазон стоимостей верный. В ячейке H2 и I2 вставим формулы «=МИН(C2:C100001))» и «=МАКС(C2:C100001))» соответственно и получим нужные числа (рис. 7).

G	H	I	J
Количество транзакций	Минимальная стоимость	Максимальная стоимость	
100000	100,01	4999,92	

Рис 7. Диапазон стоимостей

• **Даты за 2024 год:** Для выявления диапазона дат ищем минимальное и максимальное значения в столбце TransactionDate. В ячейке J2 и K2 вставим

формулы «=МИН(B2:B100001)» и «=МАКС(B2:B100001)» соответственно (рис. 8).

	I	J	K	L
ность	Максимальная стоимость	Начальная дата	Конечная дата	
100,01	4999,92	2024-01-01 00:00:00	2024-12-30 00:00:00	

Рис 8. Диапазон дат

В ходе всех проверок было доказано, что сгенерированные данные соответствуют запросу. Если бы какой-то пункт показал значение, отличное от запрошенного, в чат Deepseek нужно было бы отправить запрос с описанием несоответствия, например «В таблице присутствуют даты за 2023 год, а нужны только за 2024». В таком случае нейросеть перепишет Python код, который заново создаст таблицу.

Для проведения анализа на Python в блокноте Google Colab импортируем необходимые библиотеки (рис. 9): **pandas** – для работы с данными в таблице, **matplotlib** и **seaborn** – для визуализации результатов анализа.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Рис 9. Импортирование библиотек

Напишем функцию `calculate_rfm` (рис. 10), которая будет рассчитывать основные RFM-метрики: Recency (дни с последней покупки), Frequency (количество покупок) и Monetary (общая сумма), используя текущую дату как последнюю дату транзакции.

```
def calculate_rfm(df):
    current_date = df['TransactionDate'].max()
    print(f"\nДата анализа: {current_date.strftime('%Y-%m-%d')}")

    rfm = df.groupby('CustomerID').agg({
        'TransactionDate': lambda x: (current_date - x.max()).days,
        'CustomerID': 'count',
        'Amount': 'sum'
    }).rename(columns={
        'TransactionDate': 'Recency',
        'CustomerID': 'Frequency',
        'Amount': 'Monetary'
    }).reset_index()

    return rfm
```

Рис 10. Функция calculate_rfm

Реализуем функцию assign_scores (рис. 11), присваивающую клиентам оценки от одного до трёх для каждой метрики (Recency, Frequency, Monetary) на основе 33% и 66% квантилей, формируя комбинированную RFM-Оценку в виде строки (например, "323").

```
def assign_scores(rfm):
    metrics = ['Recency', 'Frequency', 'Monetary']
    quantiles = rfm[metrics].quantile([0.33, 0.66]).to_dict()

    def calculate_score(x, metric):
        if metric == 'Recency':
            if x <= quantiles[metric][0.33]: return 3
            elif x <= quantiles[metric][0.66]: return 2
            else: return 1
        else:
            if x <= quantiles[metric][0.33]: return 1
            elif x <= quantiles[metric][0.66]: return 2
            else: return 3

    for metric in metrics:
        rfm[f'{metric[0]}_Score'] = rfm[metric].apply(
            lambda x: calculate_score(x, metric))

    rfm['RFM_Score'] = rfm['R_Score'].astype(str) + rfm['F_Score'].astype(str) + rfm['M_Score'].astype(str)

    return rfm
```

Рис 11. Функция assign_score

Теперь в функции create_segments (рис. 12) определим сегменты клиентов по их RFM-оценкам, например, "333" — лучшие клиенты, "111" — потерянные, и присвоим каждому клиенту соответствующий сегмент.

```
def create_segments(rfm):
    segment_map = {
        '333': 'Лучшие клиенты',
        '322': 'Лояльные',
        '233': 'Почти ушедшие',
        '311': 'Новые',
        '111': 'Потерянные',
        '322|323|332': 'Перспективные',
        '233|223': 'Теряющие интерес'
    }

    rfm['Segment'] = 'Другие'
    rfm.loc[rfm['RFM_Score'] == '333', 'Segment'] = 'Лучшие клиенты'
    rfm.loc[rfm['RFM_Score'] == '322', 'Segment'] = 'Лояльные'
    rfm.loc[rfm['RFM_Score'] == '233', 'Segment'] = 'Почти ушедшие'
    rfm.loc[rfm['RFM_Score'] == '311', 'Segment'] = 'Новые'
    rfm.loc[rfm['RFM_Score'] == '111', 'Segment'] = 'Потерянные'
    rfm.loc[rfm['RFM_Score'].isin(['323', '332']), 'Segment'] = 'Перспективные'
    rfm.loc[rfm['RFM_Score'].isin(['223']), 'Segment'] = 'Теряющие интерес'

    return rfm
```

Рис 12. Функция create_segments

В функции analyze_segments (рис. 13) будем анализировать сегменты, выводя средние и медианные значения RFM-метрик, количество клиентов в каждом сегменте и их процентное распределение. Эти данные понадобятся для построения графиков.

```
def analyze_segments(rfm):
    segment_analysis = rfm.groupby('Segment').agg({
        'Recency': ['mean', 'median'],
        'Frequency': ['mean', 'median'],
        'Monetary': ['mean', 'median'],
        'CustomerID': 'count'
    }).sort_values(('CustomerID', 'count'), ascending=False)

    segment_dist = rfm['Segment'].value_counts(normalize=True) * 100
```

Рис 13. Функция analyze_segments

Создадим функцию visualize_rfm (рис. 14), в которой будут построены графики: распределение клиентов по сегментам, гистограммы Recency/Frequency/Monetary и тепловая карта медианных значений метрик для наглядного представления данных.

```
def visualize_rfm(rfm):
    plt.figure(figsize=(15, 10))

    plt.subplot(2, 2, 1)
    segment_order = rfm['Segment'].value_counts().index
    sns.countplot(y='Segment', data=rfm, order=segment_order, palette='viridis')
    plt.title('Распределение клиентов по сегментам')
    plt.xlabel('Количество клиентов')

    plt.subplot(2, 2, 2)
    sns.histplot(rfm['Recency'], bins=30, kde=True, color='skyblue')
    plt.title('Распределение Recency')
    plt.xlabel('Дней с последней покупки')

    plt.subplot(2, 2, 3)
    sns.histplot(rfm['Frequency'], bins=30, kde=True, color='salmon')
    plt.title('Распределение Frequency')
    plt.xlabel('Количество покупок')

    plt.subplot(2, 2, 4)
    sns.histplot(rfm['Monetary'], bins=30, kde=True, color='lightgreen')
    plt.title('Распределение Monetary')
    plt.xlabel('Общая сумма покупок')

    plt.tight_layout()

    plt.figure(figsize=(10, 6))
    heatmap_data = rfm.groupby('Segment')[['Recency', 'Frequency', 'Monetary']].median().loc[segment_order]
    sns.heatmap(heatmap_data, annot=True, fmt=".0f", cmap="YlGnBu",
                linewidths=.5, cbar_kws={'label': 'Медианное значение'})
    plt.title('Медианные значения RFM-метрик по сегментам')
    plt.show()
```

Рис 14. Функция visualize_rfm

Теперь остаётся только вызвать функцию для чтения файла rfm_data_2024.xlsx и вызвать все созданные выше функции (рис. 15). Самое главное – не забыть импортировать в блокнот файл, чтобы при выполнении кода не возникало ошибки о ненахождении файла для чтения.

```
data = pd.read_excel("rfm_data_2024.xlsx")
rfm = calculate_rfm(data)
rfm = assign_scores(rfm)
rfm = create_segments(rfm)
analyze_segments(rfm)
visualize_rfm(rfm)
```

Рис 15. Вызов функций

В результате выполнения кода получаем графики распределения клиентов (рис. 16) и тепловую карту метрик (рис. 17).

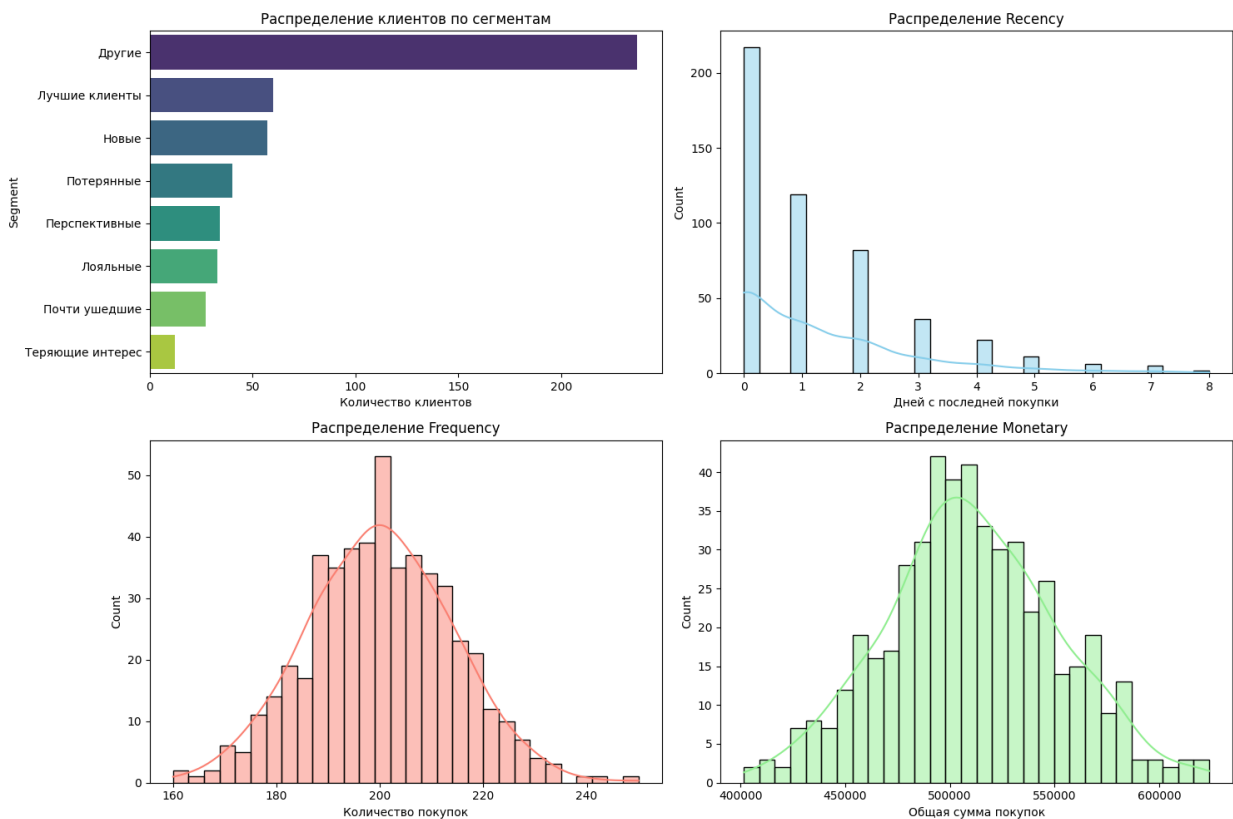


Рис 16. Графики распределения



Рис 17. Тепловая карта метрик

4 Выводы

Проведенное исследование подтвердило эффективность использования RFM-анализа на языке Python для сегментации клиентской базы. С помощью нейросети DeepSeek был успешно сгенерирован синтетический датасет, полностью соответствующий заданным требованиям: 500 уникальных

клиентов, 1000 товаров, 100000 транзакций с диапазоном стоимостей 100-5000 рублей за 2024 год. Данные прошли верификацию в Excel, что подтвердило их корректность. В ходе исследования был разработан алгоритм автоматического расчета ключевых RFM-метрик (Recency, Frequency, Monetary) с последующей классификацией клиентов по трехбалльной шкале и распределением по сегментам. Визуализация результатов анализа, включая распределение клиентов и тепловую карту метрик, наглядно продемонстрировала различия между сегментами. Применение Python и облачного сервиса Google Colab позволило эффективно обработать большой объем данных, автоматизировать расчеты и визуализацию, что подтверждает перспективность данного подхода для маркетингового анализа.

Библиографический список

1. Голубков Е. П. RFM-анализ: методика и практика применения //Маркетинг в России и за рубежом. 2013. №. 6. С. 11-24.
2. Ткаченко А. В. RFM-анализ-эффективный инструмент увеличения повторных продаж //Интернет-маркетинг. 2017. №. 2. С. 118-129.
3. Михайлов А. Н. Анализ данных в CRM: как Python помогает улучшить стратегию взаимодействия с клиентами //Вестник науки. 2024. Т. 3. №. 10 (79). С. 173-177.
4. Александров В. И. Применение RFM-анализа при разработке таргетированных маркетинговых стратегий в сфере e-commerce //Маркетинг и маркетинговые исследования. 2014. №. 5. С. 332-339.