Государственное бюджетное профессиональное
образовательное учреждение Московской области
«Физико-технический колледж»

# Аналитический отчёт

Работу выполнила:
студент группы № ИСП-22
Кривобокова Ольга Сергеевна

Долгопрудный, 2024

# ВВЕДЕНИЕ

Рынок жилья в Московском регионе считается одним из наиболее конкурентных и динамичных в России. Понимание его тенденций и факторов, определяющих цены, крайне важно для инвесторов, покупателей и продавцов. В этом исследовании мы проведем парсинг и анализ набора данных, собранного с сайта Циан, который включает объявления о продаже квартир в Москве и окрестностях, что позволит выявить ключевые паттерны и понять, какие аспекты влияют на формирование цен в данном сегменте рынка.

**Цель:** Целью данной работы является сбор и анализ информации о квартирах, предлагаемых на продажу в Московском регионе, с использованием данных с сайта Циан. Исследование направлено на выявление ключевых факторов, определяющих цены на жилье, а также на формирование комплексного представления о текущем состоянии рынка недвижимости.

**Актуальность:** Актуальность данной темы обусловлена динамичным развитием рынка недвижимости в Московском регионе, где наблюдается постоянный рост цен и изменение потребительских предпочтений. Понимание факторов, влияющих на ценообразование, необходимо как для инвесторов, так и для покупателей и продавцов. В условиях неопределенности и конкуренции на рынке, анализ данных о квартирах поможет принимать обоснованные решения и прогнозировать тенденции, что делает исследование особенно значимым для всех участников рынка.

**Задачи:**

1. **Парсинг данных:** С использованием специализированных инструментов и скриптов мы осуществим парсинг информации о квартирах на продажу с сайта Циан, собирая данные о ключевых

характеристиках, таких как цена, площадь, количество комнат, район, год постройки и другие важные параметры.

2. **Подготовка данных:** После сбора информации мы проведем этап очистки и преобразования данных, включая проверку на наличие пропусков, выбросов и других ошибок, чтобы обеспечить качество и надежность анализа.

3. **Исследовательский анализ данных (EDA):** Мы проведем исследовательский анализ данных, который включает в себя построение распределений основных характеристик, визуализацию взаимосвязей между ними и выявление факторов, оказывающих наиболее значительное влияние на целевую переменную (цену).

4. **Интерпретация результатов:** На основании полученных данных и моделей мы проанализируем результаты, выявим ключевые выводы.

# МЕТОДОЛОГИЯ

**Используемые инструменты и технологии:**

- **Библиотеки для парсинга:** cianparser и Domclick Parser — инструменты, предназначенные для автоматического извлечения данных с веб-сайтов.
- **Среда разработки для парсинга и формирования датасета:** Visual Studio Code (VS Code) с расширением Jupyter, что позволяет удобно писать и исполнять код, а также визуализировать результаты.
- **Библиотеки для анализа и обработки данных:**
    - Pandas — для работы с табличными данными, их манипуляции и обработки.
    - NumPy — для выполнения математических операций и работы с многомерными массивами.
    - Matplotlib — для создания графиков и визуализации данных.
    - Seaborn — для создания более эстетичных и информативных графиков.
    - Scikit-learn — для машинного обучения и построения моделей.
- **Среда для анализа данных и построения моделей:** Google Colaboratory, которая предоставляет возможность выполнять код на Python в облаке, не требуя установки локального окружения.
- **Инструмент для визуализации данных:** Power BI, используемый для создания интерактивных отчетов и панелей мониторинга, что позволяет эффективно представлять и анализировать данные.

# СБОР ДАННЫХ

Установка библиотеки cianparser и модификация файла page.py

Сначала мы установим библиотеку cianparser, а затем внесем изменения в файл page.py. Это позволит методу get_flats() более эффективно извлекать дополнительную информацию из объявлений о продаже квартир.

**Изменённый код файла page.py:**

```python
import bs4
import re
import time
import random


class FlatPageParser:
    def __init__(self, session, url):
        self.session = session
        self.url = url

    def __load_page__(self):
        res = self.session.get(self.url)
        if res.status_code == 429:
            time.sleep(10)
        res.raise_for_status()
        self.offer_page_html = res.text
        self.offer_page_soup = bs4.BeautifulSoup(self.offer_page_html, 'html.parser')

    def __parse_flat_offer_page_json__(self):
        page_data = {
            "year_of_construction": -1,
            "object_type": -1,
            "have_loggia": -1,
            "parking_type": -1,
            "house_material_type": -1,
            "heating_type": -1,
            "finish_type": -1,
            "living_meters": -1,
            "kitchen_meters": -1,
            "floor": -1,
            "floors_count": -1,
            "phone": "",
        }
        ot = self.offer_page_soup.select_one('[data-name="OfferSummaryInfoItem"] p:nth-of-type(2)').get_text()
        page_data["object_type"] = ot
        time.sleep(5 + random.uniform(0, 5))

        pt_elements = self.offer_page_soup.select('[data-name="OfferSummaryInfoItem"] p')
        for i, p_element in enumerate(pt_elements):
            if "Парковка" in p_element.get_text():
                parking_type_element = pt_elements[i + 1]
                print(i)
                page_data["parking_type"] = parking_type_element.get_text()
                time.sleep(5 + random.uniform(0, 5))
                break
```

```
    else:
        page_data["parking_type"] = -1

    hl_elements = self.offer_page_soup.select('[data-name="OfferSummaryInfoItem"] p')
    for i, hl_element in enumerate(hl_elements):
        if "Балкон/лоджия" in hl_element.get_text():
            have_loggia_element = hl_elements[i + 1]
            print(i)
            page_data["have_loggia"] = have_loggia_element.get_text()
            time.sleep(5 + random.uniform(0, 5))
            break
    else:
        page_data["have_loggia"] = -1
    ch_elements = self.offer_page_soup.select('[data-name="OfferSummaryInfoItem"] p')
    for i, ch_element in enumerate(ch_elements):
        if "Высота потолков" in ch_element.get_text():
            ceiling_height_element = ch_elements[i + 1]
            print(i)
            page_data["ceiling_height"] = ceiling_height_element.get_text()
            time.sleep(5 + random.uniform(0, 5))
            break
    else:
        page_data["ceiling_height"] = -1
    spans = self.offer_page_soup.select("span")
    for index, span in enumerate(spans):

        if "Тип дома" == span.text:
            page_data["house_material_type"] = spans[index + 1].text
            time.sleep(5 + random.uniform(0, 5))
        if "Отделка" == span.text:
            page_data["finish_type"] = spans[index + 1].text
            time.sleep(5 + random.uniform(0, 5))
        if "Площадь кухни" == span.text:
            page_data["kitchen_meters"] = spans[index + 1].text
            time.sleep(5 + random.uniform(0, 5))
        if "Жилая площадь" == span.text:
            page_data["living_meters"] = spans[index + 1].text
            time.sleep(5 + random.uniform(0, 5))
        if "Год постройки" in span.text:
            page_data["year_of_construction"] = spans[index + 1].text
            time.sleep(5 + random.uniform(0, 5))
        if "Год сдачи" in span.text:
            page_data["year_of_construction"] = spans[index + 1].text
            time.sleep(5 + random.uniform(0, 5))
        if "Этаж" == span.text:
            ints = re.findall(r'\d+', spans[index + 1].text)
            if len(ints) == 2:
                page_data["floor"] = int(ints[0])
                page_data["floors_count"] = int(ints[1])
            time.sleep(5 + random.uniform(0, 5))

    if "+7" in self.offer_page_html:
        page_data["phone"] = self.offer_page_html[self.offer_page_html.find("+7"): self.offer_page_html.find("+7") + 16].split('"')[0]. \
            replace(" ", ""). \
            replace("-", "")
        time.sleep(5 + random.uniform(0, 5))
    return page_data
def parse_page(self):
    self.__load_page__()
    return self.__parse_flat_offer_page_json__()
```

# АНАЛИЗ ДАННЫХ

## 1. Введение

В данном отчете представлена аналитика данных о квартирах, собранных с веб-сайта Cian. Цель анализа заключается в очистке и подготовке данных, а также в исследовании факторов, влияющих на цену квадратного метра.

## 2. Предварительный анализ данных

На начальном этапе был проведен предварительный анализ данных, включающий изучение первых строк датафрейма и подсчет количества дубликатов. Выяснилось, что в наборе данных есть дубликаты, которые были удалены для повышения качества анализа.

## 3. Очистка данных

В процессе очистки данных были удалены ненужные колонки, такие как автор и тип автора, которые не влияют на стоимость квартир. Также была представлена общая информация о наборе данных, включая количество пропущенных значений. Визуализация пропусков с помощью тепловой карты помогла лучше понять их распределение.

Для дальнейшего анализа значения -1 были заменены на NaN, что позволило выявить и удалить строки с критическими пропусками в важнейших переменных, таких как количество комнат и цены.

## 4. Преобразование данных

После очистки данных было проведено преобразование типов для ключевых колонок. Например, высота потолков и площадь были преобразованы в числовые значения. Это необходимо для дальнейшего анализа и визуализации данных.

## 5. Выявление и удаление выбросов

Для выявления выбросов в данных были построены диаграммы размаха для ключевых параметров, таких как цена, общая площадь и высота потолков. Выбросы были удалены на основании визуального анализа, что позволило улучшить качество анализа.

**6. Анализ корреляций**

Была построена матрица корреляции, которая позволила выявить связи между различными параметрами. Это полезный инструмент для понимания того, как различные факторы могут влиять на цену квартир.

**7. Расчет цены за квадратный метр**

Цены за квадратный метр были рассчитаны и проанализированы с использованием гистограммы. Это позволило выявить распределение цен в зависимости от количества комнат и других факторов. Визуализация показала, как цена за квадратный метр варьируется в зависимости от характеристик квартир.

**8. Заключение**

Анализ показал, что данные содержат важную информацию, которая может быть использована для оценки стоимости квартир. Были выявлены ключевые факторы, влияющие на цену, такие как этаж, высота потолков и общая площадь. Результаты данного анализа могут быть полезны для дальнейших исследований и для потенциальных инвесторов на рынке недвижимости.

# АНАЛИТИКА

Для начала анализа я импортировала библиотеки, выгрузила файл csv, просмотрела первые 5 строк файла. Далее была работа с дубликатами, подсчитали количество, удалили и посмотрели сколько осталось. Также удалили не нужные колонки: имя автора и кем является. Смотрим количество пропущенных значений и видим пропуски:

Price, district, street, house_number, underground, residential_complex

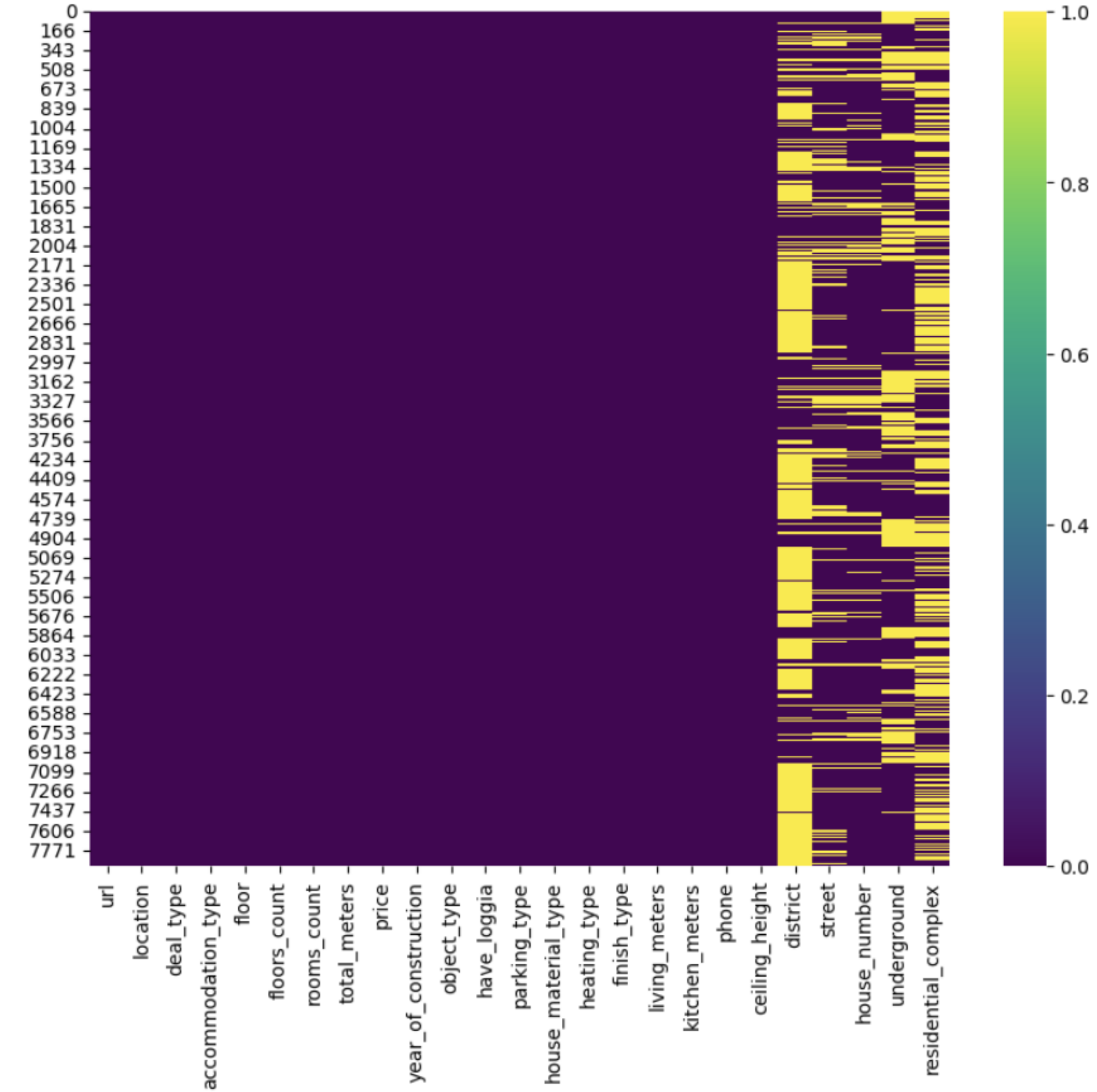
```
[341] # Количество дубликатов
      duplicates_count = df.duplicated().sum()
      print(f"Количество дубликатов: {duplicates_count}")

      # Удаление дубликатов
      df = df.drop_duplicates()
      print(f"Данные после удаления дубликатов: {df.shape}")
```

```
Количество дубликатов: 676
Данные после удаления дубликатов: (7229, 27)
```

```
[342] # Удаляем не нужные значения
      df.drop(['author', 'author_type'], axis=1, inplace=True, errors='ignore')
```

| index | 0 |
|---|---|
| 0 url | 0 |
| 1 location | 0 |
| 2 deal_type | 0 |
| 3 accommodation_type | 0 |
| 4 floor | 0 |
| 5 floors_count | 0 |
| 6 rooms_count | 0 |
| 7 total_meters | 0 |
| 8 price | 13 |
| 9 year_of_construction | 0 |
| 10 object_type | 0 |
| 11 have_loggia | 0 |
| 12 parking_type | 0 |
| 13 house_material_type | 0 |
| 14 heating_type | 0 |
| 15 finish_type | 0 |
| 16 living_meters | 0 |
| 17 kitchen_meters | 0 |
| 18 phone | 0 |
| 19 ceiling_height | 0 |
| 20 district | 4355 |
| 21 street | 1498 |
| 22 house_number | 1084 |
| 23 underground | 2207 |
| 24 residential_complex | 3585 |

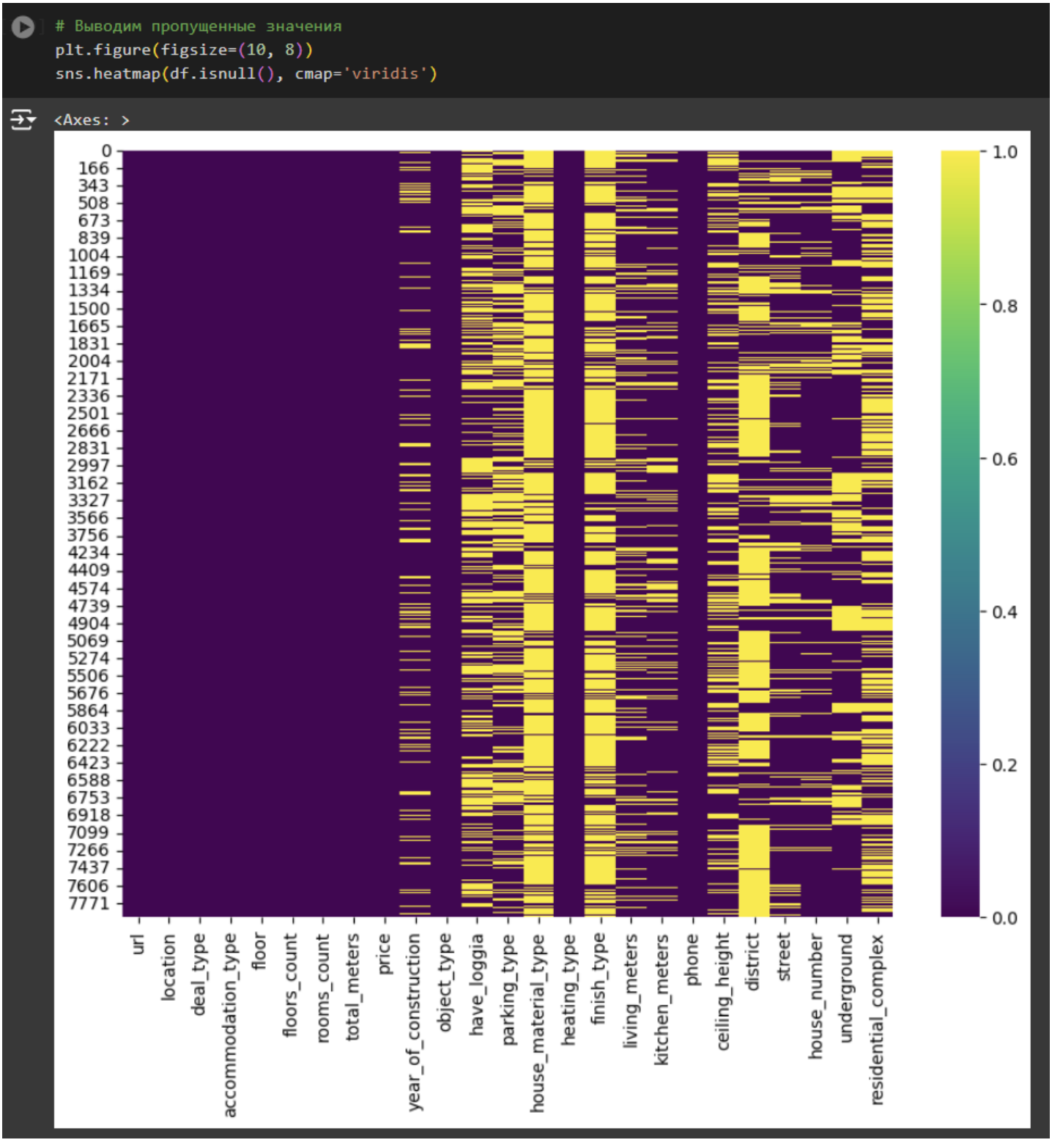
Выводим пропущенные значения в тепловой карте:



После визуализации мы меняем все -1 на NaN, чтобы посмотреть все пропущенные значения. Видим, как сильно увеличилось количество пропусков.

```
# Меняем все -1 на NaN, чтобы посмотреть пропущенные значения
df.replace('-1', np.nan, inplace=True)
```

```
# Выводим пропущенные значения
plt.figure(figsize=(10, 8))
sns.heatmap(df.isnull(), cmap='viridis')
```



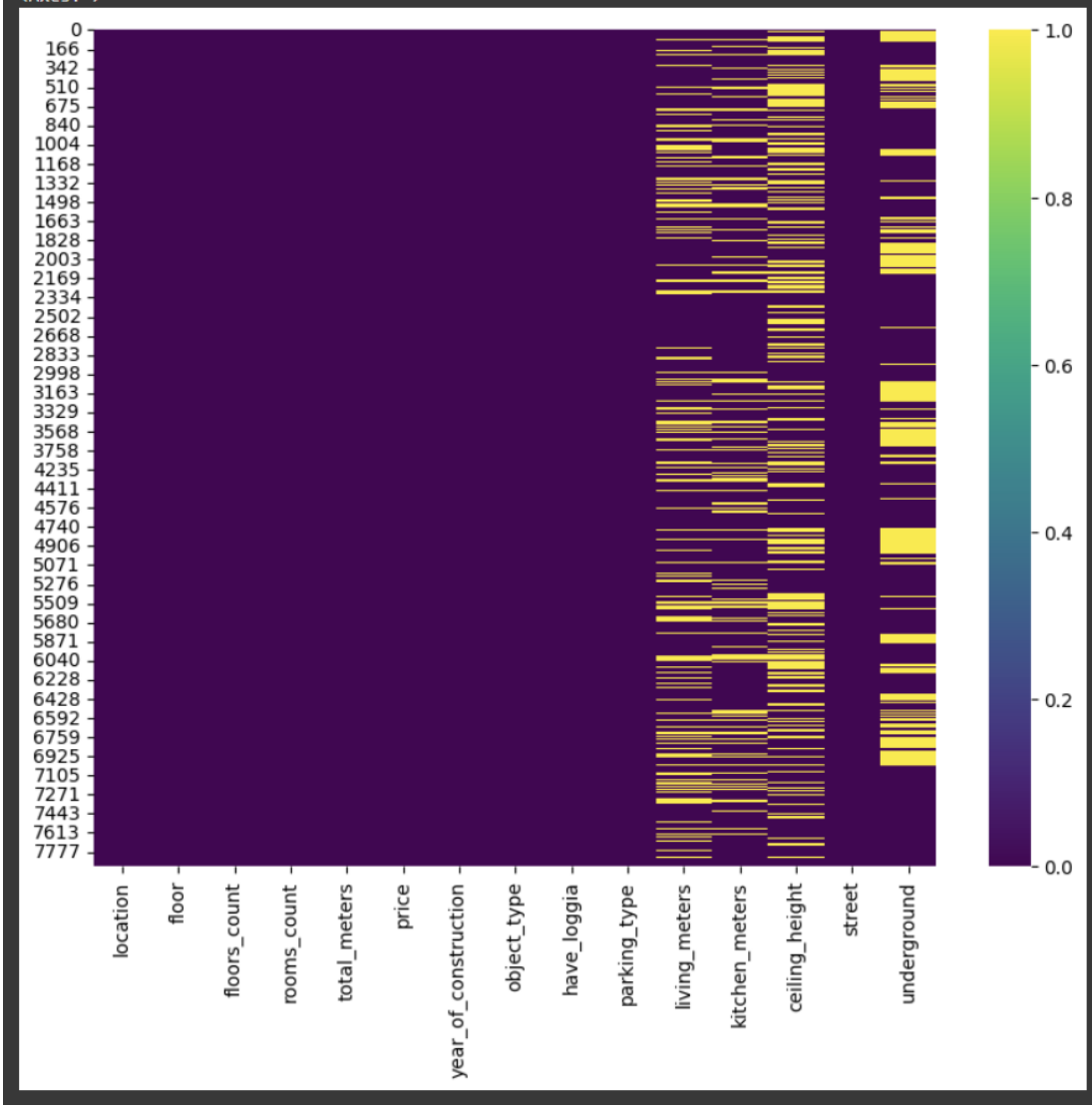
Удаляем ненужные для анализа колонки и удаляем строчки где пропущено кол-во комнат. После этого мы используем метод assign для обновления значений. Заменяем пропуски на 0, не указано.

```
[350] # Удаляем ненужные для анализа колонки
      df.drop(['heating_type', 'house_material_type', 'house_number', 'residential_complex', 'phone', 'district',
               'finish_type', 'url', 'deal_type', 'accommodation_type' ], axis=1,inplace=True, errors='ignore')
      # удаляем  строчки где пропущено кол-во комнат и цены
      df = df.dropna(subset=[ 'rooms_count'])
      df = df.query("price != 0")
```

```
[351] # Используем метод assign для обновления значений
      df = df.assign(
          year_of_construction=df['year_of_construction'].fillna(0),
          price=df['price'].fillna(0),
          street=df['street'].fillna("не указано"),
          have_loggia = df['have_loggia'].fillna('0'),
          parking_type = df['parking_type'].fillna('0')
      )
```

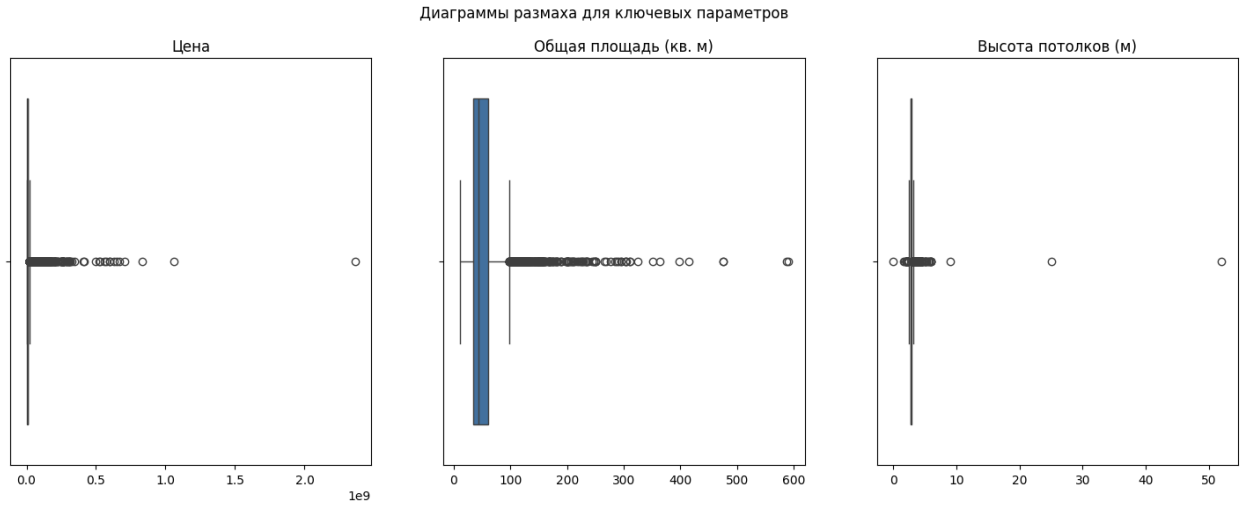
```
# Выводим пропущенные значения
plt.figure(figsize=(10, 8))
sns.heatmap(df.isnull(), cmap='viridis')
```

```
<Axes: >
```



Меняем тип данных у колонок : rooms_count, price, floor, floors_count, total_meters, living_meters, ceiling_height, year_of_construction, kitchen_meters. Пишем функцию для замены значений, чтобы осталась только новостройка и вторичка. После всех проделанных действий, производим визуализацию выбросов для цены, общую площадь, высота потолков.

```
#Делаем визуализацию выбросов
fig, axes = plt.subplots(1, 3, figsize=(18, 6))
# Price
sns.boxplot(x=df['price'], ax=axes[0]) #график будет отрисован на первой оси
axes[0].set_title('Цена') #заголовок для первого подграфика
axes[0].set_xlabel('') #подпись для оси X

# Total meters
sns.boxplot(x=df['total_meters'], ax=axes[1])
axes[1].set_title('Общая площадь (кв. м)')
axes[1].set_xlabel('')

# Ceiling height
sns.boxplot(x=df['ceiling_height'], ax=axes[2])
axes[2].set_title('Высота потолков (м)')
axes[2].set_xlabel('')

plt.suptitle('Диаграммы размаха для ключевых параметров')
plt.show()
```



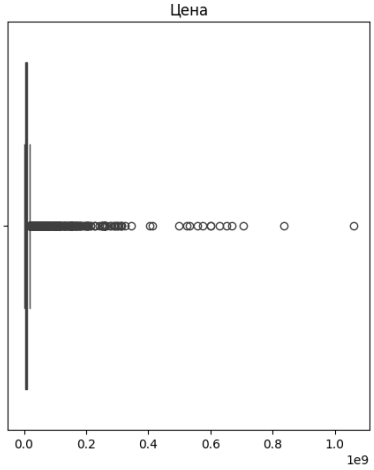
Выводим 10 максимальных и минимальных значений у цены, общей площади высоты потолков , жилая площадь, этажи, количество этажей. Удаляем аномальные значения и смотрим еще раз графики.

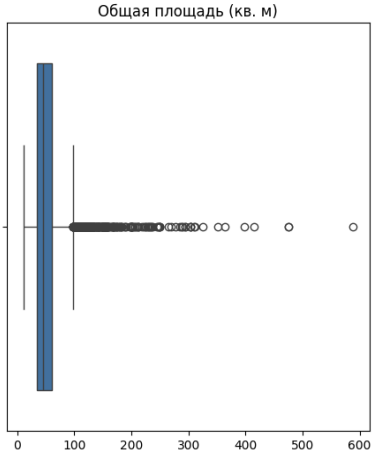
```
# 10 максимальных значений высоты потолков
top_10_ceiling_heights = df['ceiling_height'].nlargest(10)
print("10 максимальных значений высоты потолков:")
print(top_10_ceiling_heights)
```

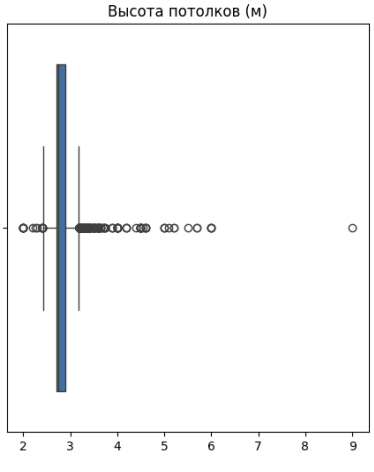
```
10 максимальных значений высоты потолков:
2250    52.0
4904    25.0
5076     9.0
225      6.0
938      6.0
5539     6.0
5643     6.0
939      5.7
3061     5.7
5515     5.5
Name: ceiling_height, dtype: float64
```

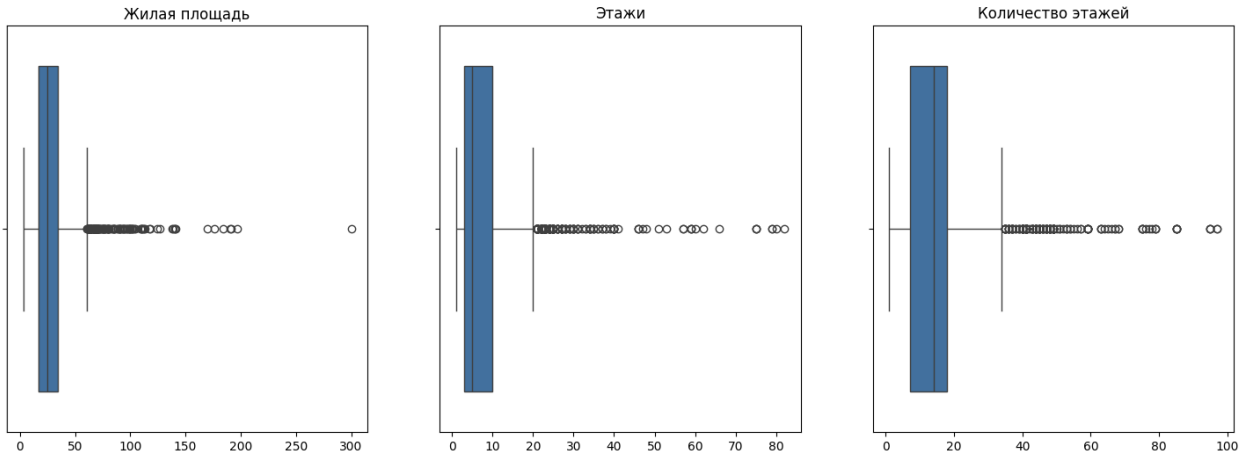
```
] # 10 минимальных значений высоты потолков
bottom_10_ceiling_heights = df['ceiling_height'].nsmallest(10)
print("10 минимальных значений высоты потолков:")
print(bottom_10_ceiling_heights)
```

```
10 минимальных значений высоты потолков:
3739    0.00
2649    1.65
1244    1.80
731     2.00
1066    2.00
2927    2.00
3722    2.00
5130    2.00
5954    2.00
6906    2.00
```



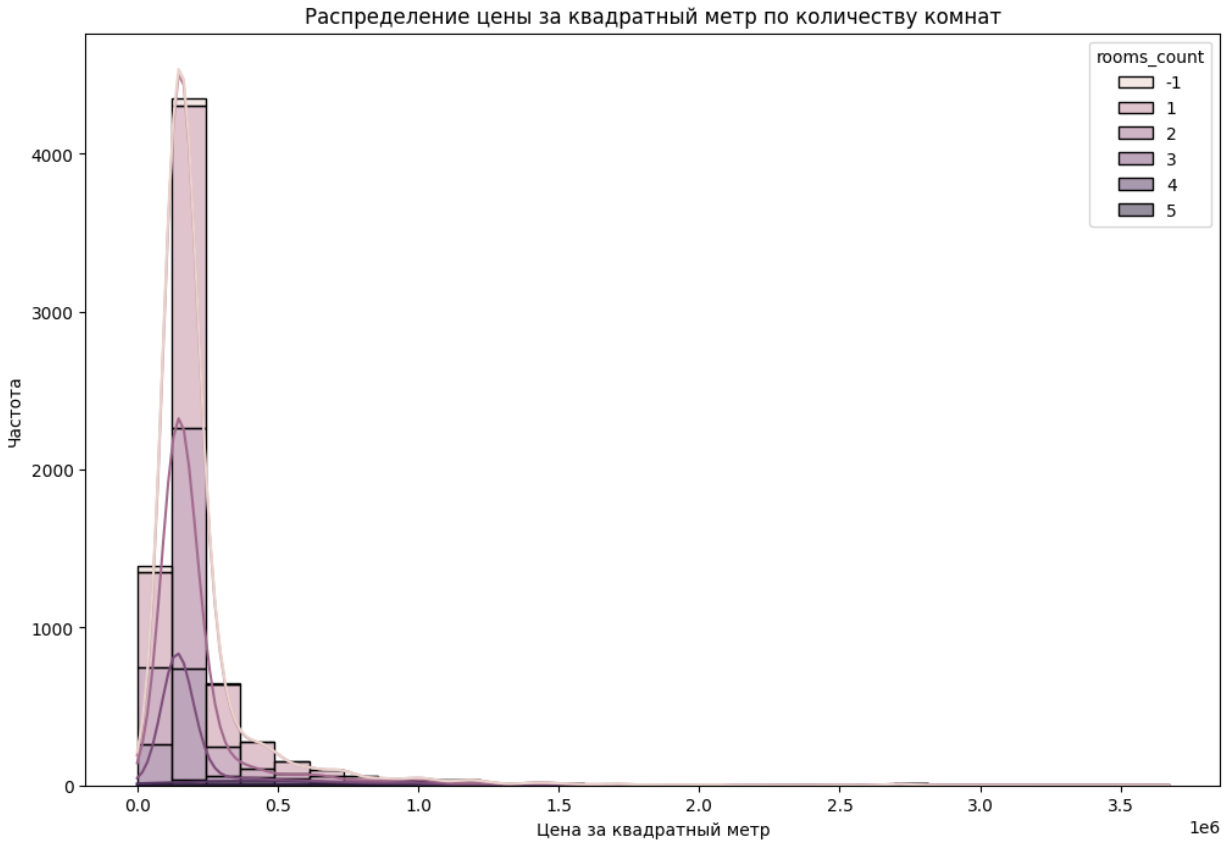






**Расчет цены за квадратный метр:**

Строим гистограмму. Делаем вывод что, цена за квадратный метр увеличивается с ростом количества комнат. Есть значительное количество выбросов, особенно в диапазоне от 1 до 3 комнат, что может свидетельствовать о наличии дорогих квартир с низким количеством комнат или наоборот.

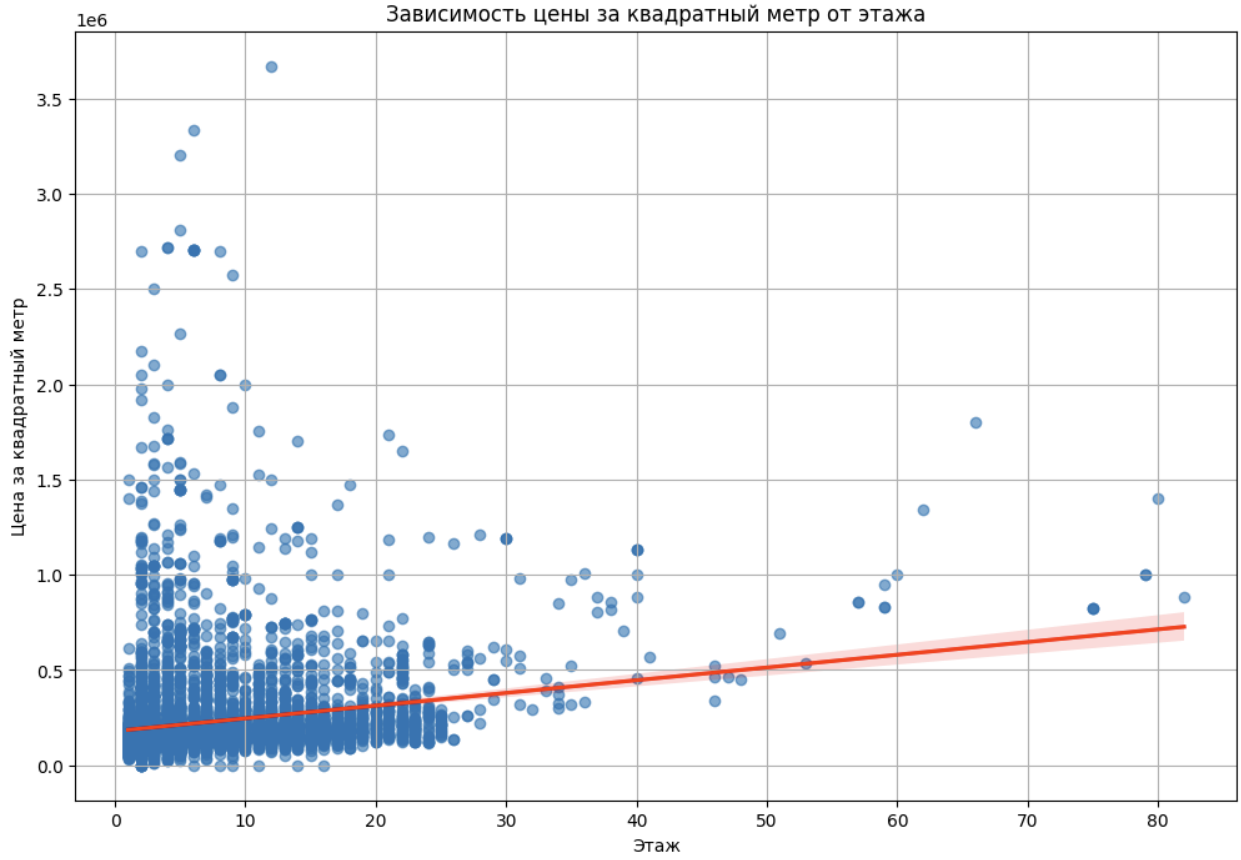
```
# Строим гистограмму с разбивкой по количеству комнат (rooms_count)
plt.figure(figsize=(12, 8))
sns.histplot(data=df, x='price_per_sqm', hue='rooms_count', multiple='stack', bins=30, kde=True)
plt.title('Распределение цены за квадратный метр по количеству комнат')
plt.xlabel('Цена за квадратный метр')
plt.ylabel('Частота')
plt.show()
```



**Зависимость цены за квадратный метр от этажа:**

На графике видно, что цена за квадратный метр в основном распределена в диапазоне от 0 до 1.5 миллиона рублей.

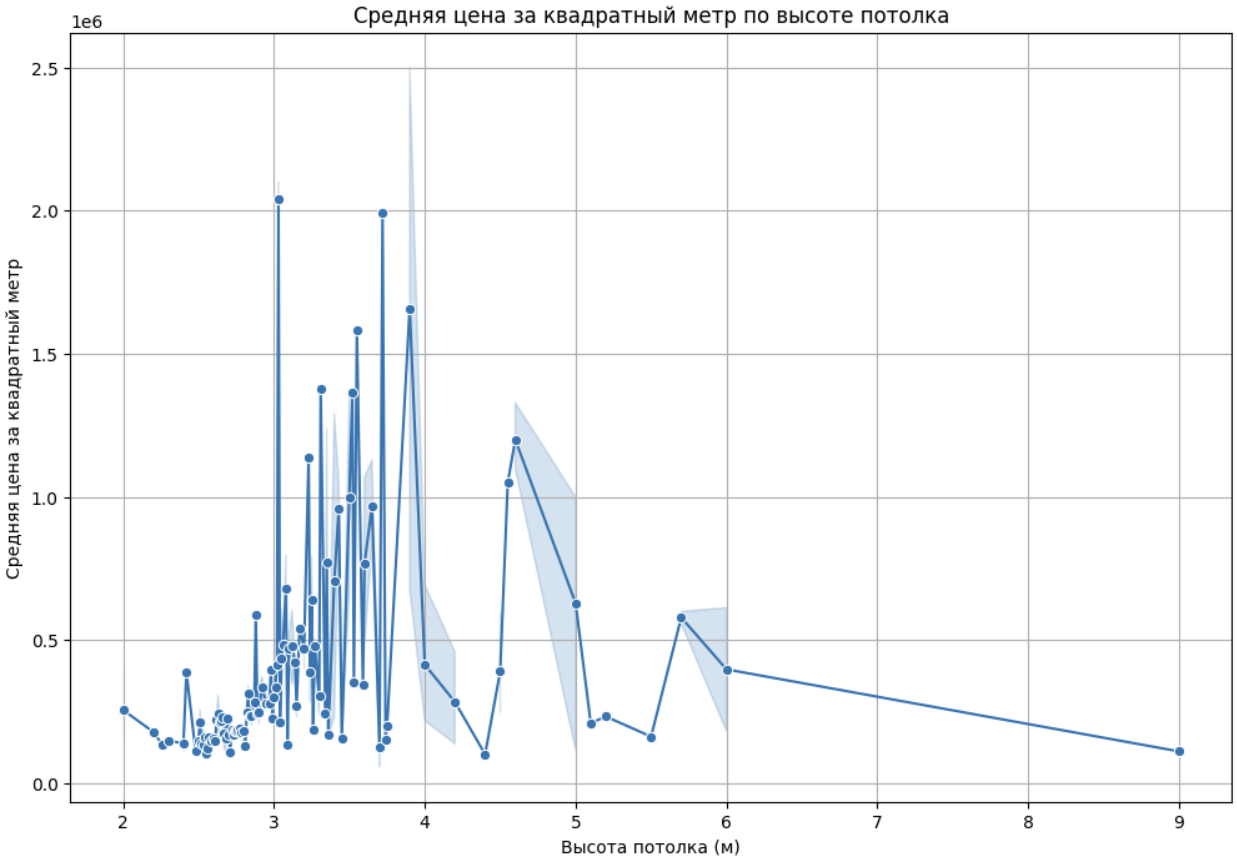
```
#Создаем график регрессии для столбцов 'floor' и 'price_per_sqm'
plt.figure(figsize=(12, 8))
sns.regplot(data=df, x='floor', y='price_per_sqm', scatter_kws={'alpha':0.6}, line_kws={'color':'red'})
plt.title('Зависимость цены за квадратный метр от этажа')
plt.xlabel('Этаж')
plt.ylabel('Цена за квадратный метр')
plt.grid(True)
plt.show()
```



**Зависимость цены за квадратный метр от высоты потолка:**

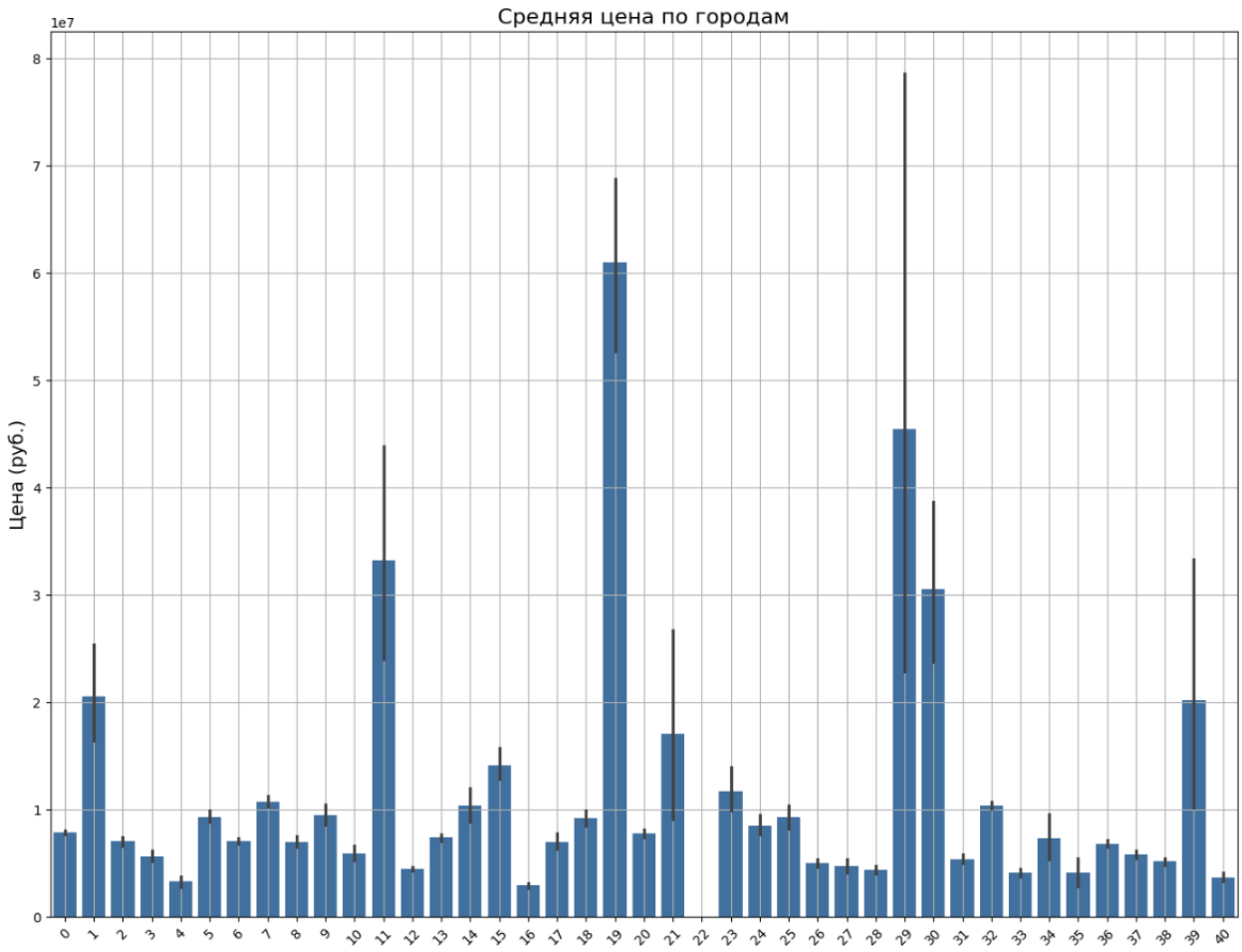
График показывает, что существует небольшая положительная корреляция между высотой потолка и ценой за квадратный метр. Хотя основная масса данных сосредоточена в диапазоне высоты потолка от 2.5 до 3.5 метров, есть некоторые выбросы с высокими значениями, которые могут указывать на элитные квартиры.

```
# Строим линейный график зависимости средней цены за квадратный метр от высоты потолка
plt.figure(figsize=(12, 8))
sns.lineplot(data=df, x='ceiling_height', y='price_per_sqm', estimator='mean', marker='o')
plt.title('Средняя цена за квадратный метр по высоте потолка')
plt.xlabel('Высота потолка (м)')
plt.ylabel('Средняя цена за квадратный метр')
plt.grid(True)
plt.show()
```



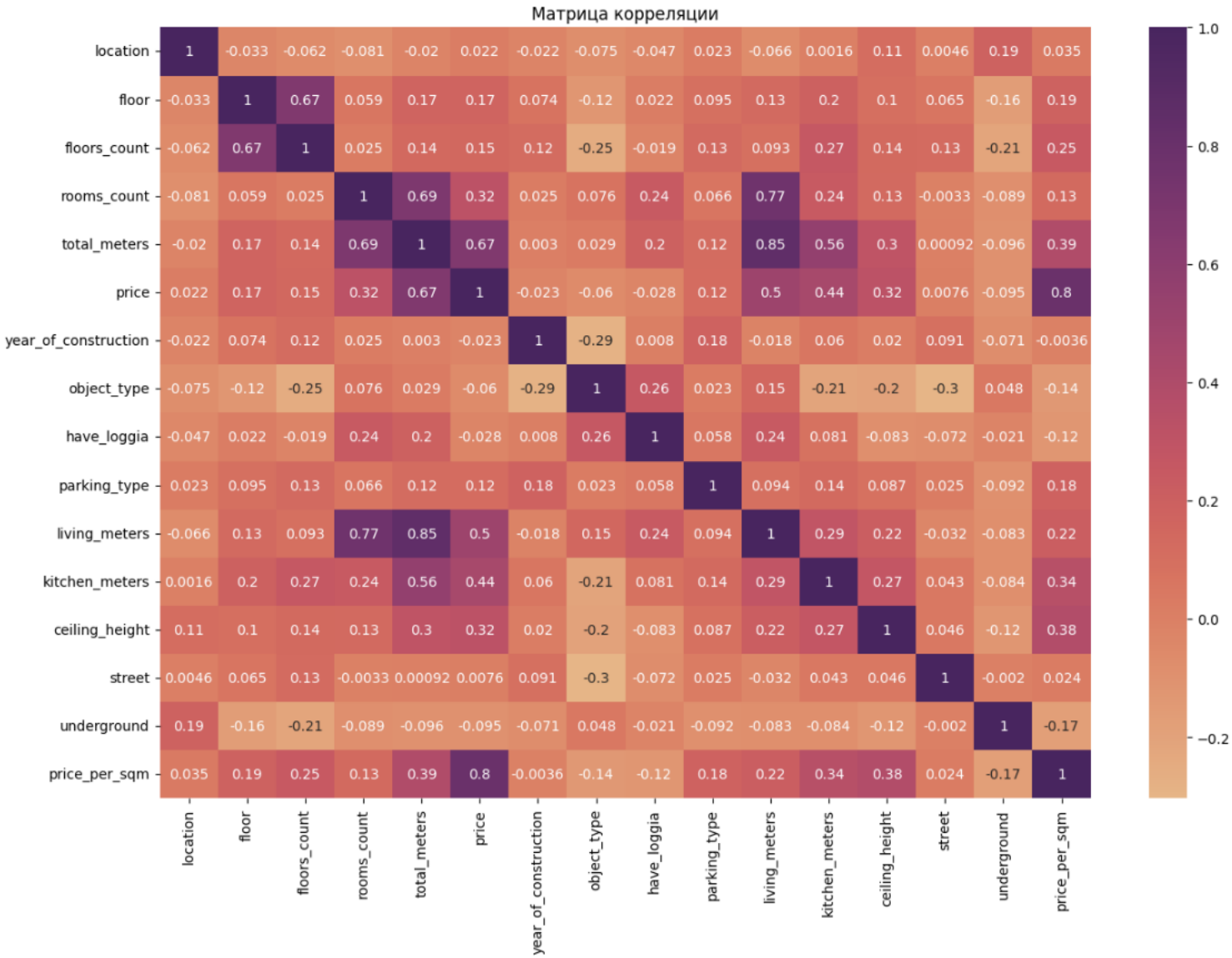
Цена по городам: Самые больше значения мы видим в городах: Москва(19) Талодом(29) Ивантеевка(11) Троицк(30)

```
# Строим столбчатую диаграмму для отображения средней цены по каждому городу

plt.figure(figsize=(16, 12))
sns.barplot(data=df, x='location', y='price', estimator='mean')
plt.title('Средняя цена по городам', fontsize=16)
plt.xlabel('Город', fontsize=14)
plt.ylabel('Цена (руб.)', fontsize=14)
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```

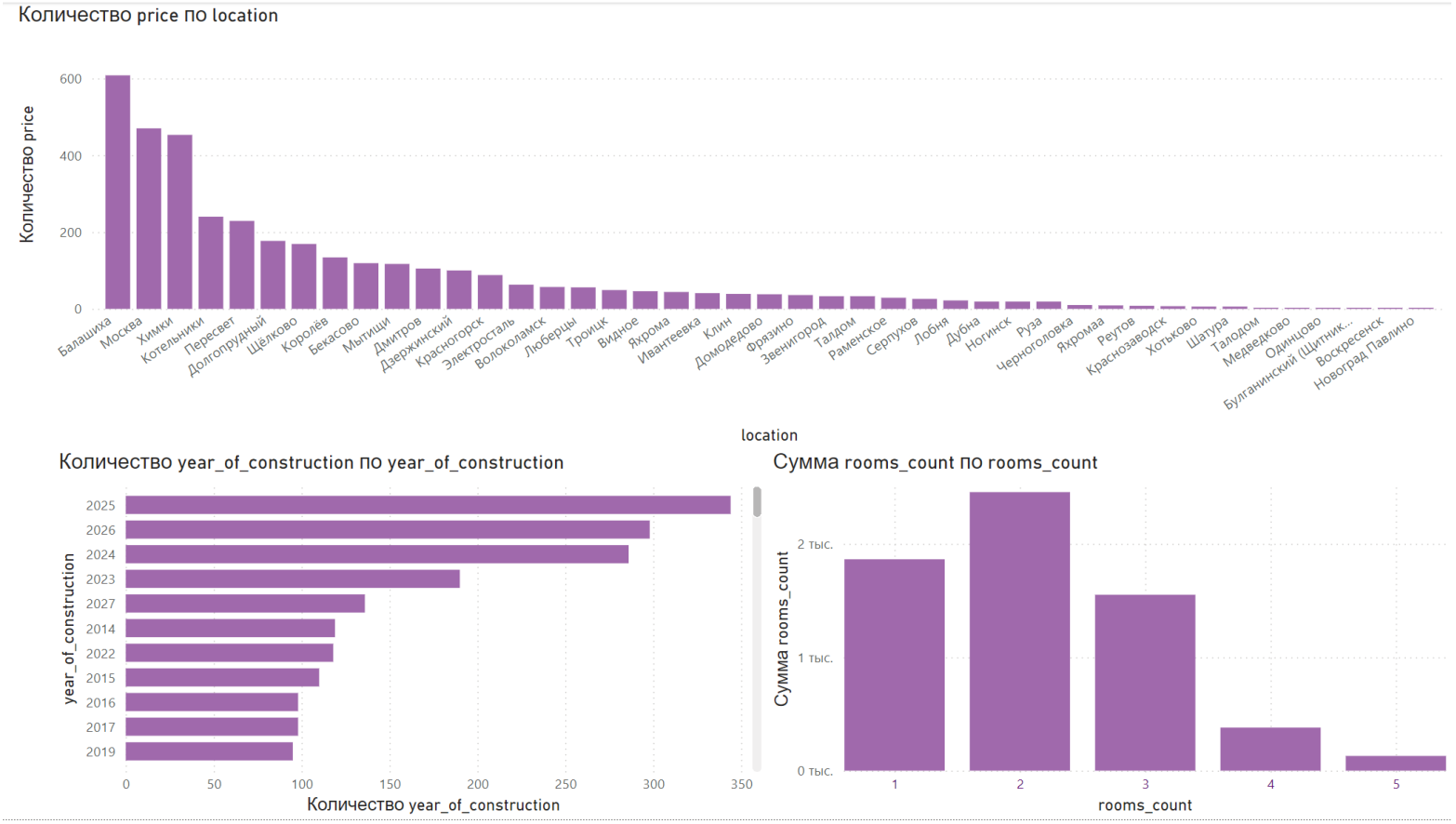


**Вывод по матрице корреляции:**

Стоимость зависит: От этажа, этажности дома, года постройки, количества комнат, общей площади квартиры, площади кухни и жилой площади, высоты потолка и от стоимости за квадратный метр. И небольшая зависимость от парковки. Стоимость за квадратный метр зависит: От этажа, этажности дома, цены, площади (общая площадь, площадь кухни и жилая), года постройки здания (больше чем общая стоимость квартиры), высоты потолка + от наличия парковки.



# Power BI


Количество price по location
Количество price
600
400
200
0
Балашиха
Москва
Химки
Котельники
Пересвет
Долгопрудный
Щёлково
Королёв
Бекасово
Мытищи
Дмитров
Дзержинский
Красногорск
Электросталь
Волоколамск
Люберцы
Троицк
Видное
Якрома
Ивантеевка
Клин
Домодедово
Фрязино
Звенигород
Талдом
Раменское
Серпухов
Лобня
Дубна
Ногинск
Руза
Черноголовка
Яхромаа
Реутов
Краснозаводск
Хотьково
Шатура
Талдом
Медведково
Одинцово
Булганинский (Щитник...
Воскресенск
Новоград Павлино
location
Количество year_of_construction по year_of_construction
year_of_construction
2025
2026
2024
2023
2027
2014
2022
2015
2016
2017
2019
0
50
100
150
200
250
300
350
Количество year_of_construction
Сумма rooms_count по rooms_count
Сумма rooms_count
2 тыс.
1 тыс.
0 тыс.
1
2
3
4
5
rooms_count