

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

Аналитический отчет

Работу выполнил:
Студент группы № ИСП-21
Кузнецов Илья
Проверил:
преподаватель информатики
Базяк Г.В.

Долгопрудный, 2024

Введение

Рынок недвижимости в Московском регионе является одним из самых динамичных и конкурентных в России. Понимание его тенденций и факторов, влияющих на цены, является ключевым для инвесторов, покупателей и продавцов. В данной работе мы рассмотрим парсинг и анализ датасета, собранного с сайта Циан, охватывающего объявления о продаже квартир в Москве и Московской области.

Цель: Целью данной работы является сбор и анализ данных о квартирах на продажу в Московском регионе с использованием сайта Циан. Анализ данных позволит нам понять ключевые факторы, влияющие на цены, и получить ценную информацию о рынке недвижимости.

Задачи:

1. Парсинг данных: с помощью соответствующих инструментов и скриптов мы произведем парсинг данных о квартирах на продажу с сайта Циан, собирая информацию о ключевых параметрах, таких как цена, площадь, количество комнат, район, год постройки и другие;
2. Подготовка данных: после сбора данных мы проведем очистку и преобразование данных, проверяя на пропуски, выбросы и ошибки;
3. Исследовательский анализ данных (EDA): Мы выполним исследовательский анализ данных, включающий построение распределения основных параметров, визуализацию взаимосвязей между ними, а также определение признаков, оказывающих наиболее сильное влияние на целевую переменную (цену).

Основная часть

1. Методология

Для сбора, обработки и аналитики данных я использовал следующие инструменты:

Сбор данных:

- Циан-парсер: Используемый инструмент для автоматического сбора данных с сайта Циан.

Очистка и классификация данных с помощью библиотек Python:

- pandas: для работы с таблицами данных, манипулирования, очистки и преобразования данных;
- numpy: для работы с многомерными массивами, математическими операциями и статистическими расчетами;
- seaborn: для визуализации данных, создания красивых и информативных графиков.

Анализ и построение диаграмм/графиков:

- pandas: для анализа данных, группировки, агрегации и работы с временными рядами;
- numpy: для выполнения математических расчетов и статистических операций;
- seaborn: для визуализации данных и создания привлекательных графиков;
- matplotlib: для создания более гибких и настраиваемых графиков;
- Power BI: Мощный инструмент для визуализации данных, создания интерактивных отчетов и dashboards.

2. Обработка данных

Перед анализом я выполнил обработку и очистку данных.

Удаление дубликатов:

```
[ ] df['url'].value_counts()[:5]
```

| url | count |
|--|-------|
| https://lyubertsy.cian.ru/sale/flat/306254042/ | 10 |
| https://korolev.cian.ru/sale/flat/299371063/ | 10 |
| https://vidnoye.cian.ru/sale/flat/307996713/ | 8 |
| https://lobnya.cian.ru/sale/flat/307641328/ | 8 |
| https://www.cian.ru/sale/flat/301450189/ | 8 |

dtype: int64

```
df.drop_duplicates(subset='url', inplace=True)  
df['url'].value_counts()[:5]
```

| url | count |
|---|-------|
| https://www.cian.ru/sale/flat/308167237/ | 1 |
| https://lobnya.cian.ru/sale/flat/308575050/ | 1 |
| https://klin.cian.ru/sale/flat/303065081/ | 1 |
| https://klin.cian.ru/sale/flat/307588215/ | 1 |
| https://klin.cian.ru/sale/flat/303624087/ | 1 |

dtype: int64

Замена всех некорректных значений на пустые и заполнение некоторых пропусков данными:

```
[ ] df['living_meters'] = df['living_meters'].str.replace(r'(\s.*)', '', regex=True).replace(',', '.', regex=True).astype(float)
    df['kitchen_meters'] = df['kitchen_meters'].str.replace(r'(\s.*)', '', regex=True).replace(',', '.', regex=True).astype(float)

df.dropna(subset=['total_meters', 'year_of_construction'], inplace=True)

def replace_invalid_decimal(value):
    try:
        float(value)
        return value
    except:
        return str(value.split('.')[0]) + '.0'

df['total_meters'] = df['total_meters'].apply(lambda x: replace_invalid_decimal(x)).astype(float)

def del_str(value):
    try:
        if int(value) <= 2024:
            return value
        else:
            return np.nan
    except:
        return np.nan

df['year_of_construction'] = df['year_of_construction'].apply(lambda x: del_str(x))
```

Я заметил, что некоторые данные некорректны и при классификации выдают ошибку, поэтому начал это исправлять для дальнейшей классификации данных и их корректности.

В колонках living_meters и kitchen_meters была преписка «м²», поэтому я избавился от этой преписки с помощью регулярного выражения и классифицировал столбцы типом float.

В колонке total_meters я заметил, что некоторые значения типа float имеют после точки какие-либо символы, поэтому я решил убрать эти символы и округлить данные до целого числа, а после привел столбец к типу float.

Столбец year_of_construction нужно было отфильтровать все строки со значениями выше 2024 года, поэтому я заменил все эти значения пустыми, чтобы в дальнейшем удалить их вместе с остальными строками с пустыми значениями.

```
[ ] df['living_meters'] = df['living_meters'].fillna(df['total_meters']-df['kitchen_meters'])
    df['kitchen_meters'] = df['kitchen_meters'].fillna(df['total_meters']-df['living_meters'])
```

Далее я заполнил значения living_meters и kitchen_meters путем их вычитания из столбца total_meters для того чтобы не терять большую часть данных.

```
[ ] df['rooms_count'].replace(-1, 0, inplace=True)

df.replace(-1, np.nan, inplace=True)
df.replace('-1', np.nan, inplace=True)
```

Я заметил, что большинство данных полученных с Циан вместо пустых значений имеют значение «-1», поэтому я заменил все значения «-1» на пустую строку (кроме колонки rooms_count). В колонке rooms_count я заменил все значения «-1» на значение «0», так как в этой колонке значением «-1» обозначаются студии.

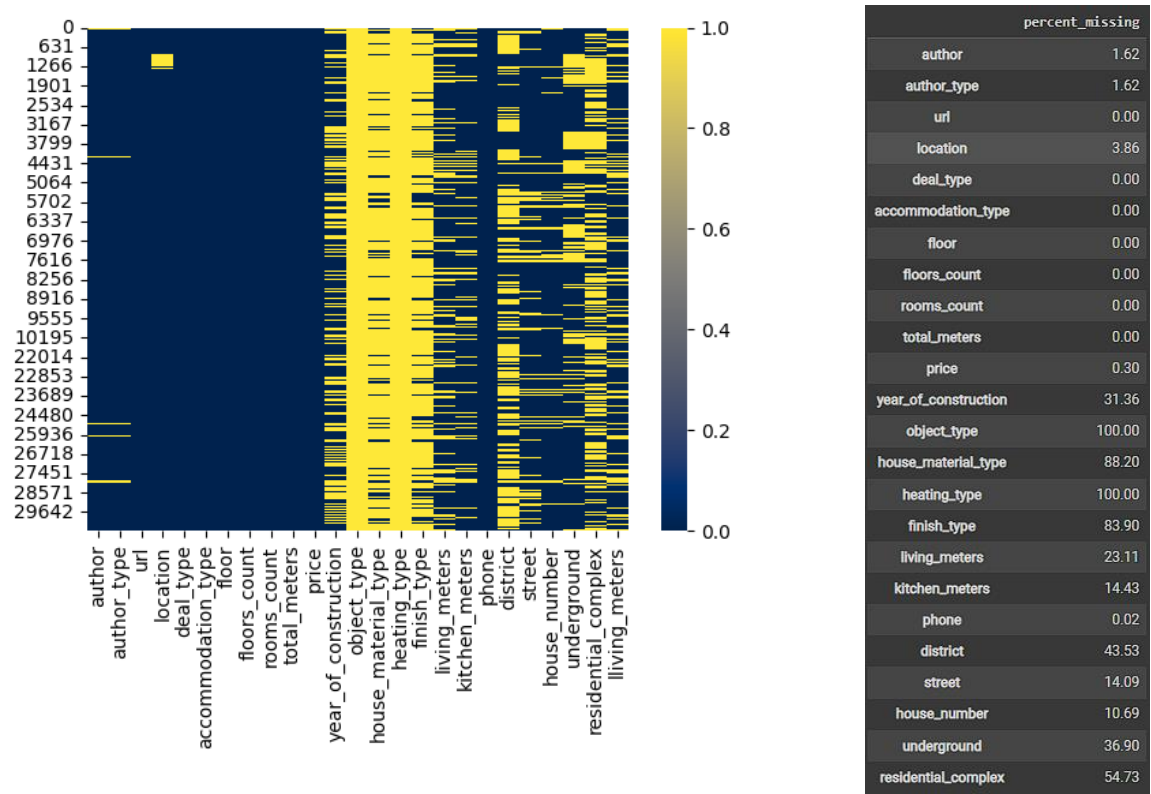
Удаление колонок, которые не будут использованы в анализе или в которых большое кол-во пустых значений:

```
[ ] df.drop(['residential_complex', 'district', 'url', 'accommodation_type',
            'phone', 'underground', 'house_number', 'deal_type', 'object_type',
            'heating_type', 'house_material_type', 'finish_type', 'street'], axis = 1, inplace = True)
df.head()
```

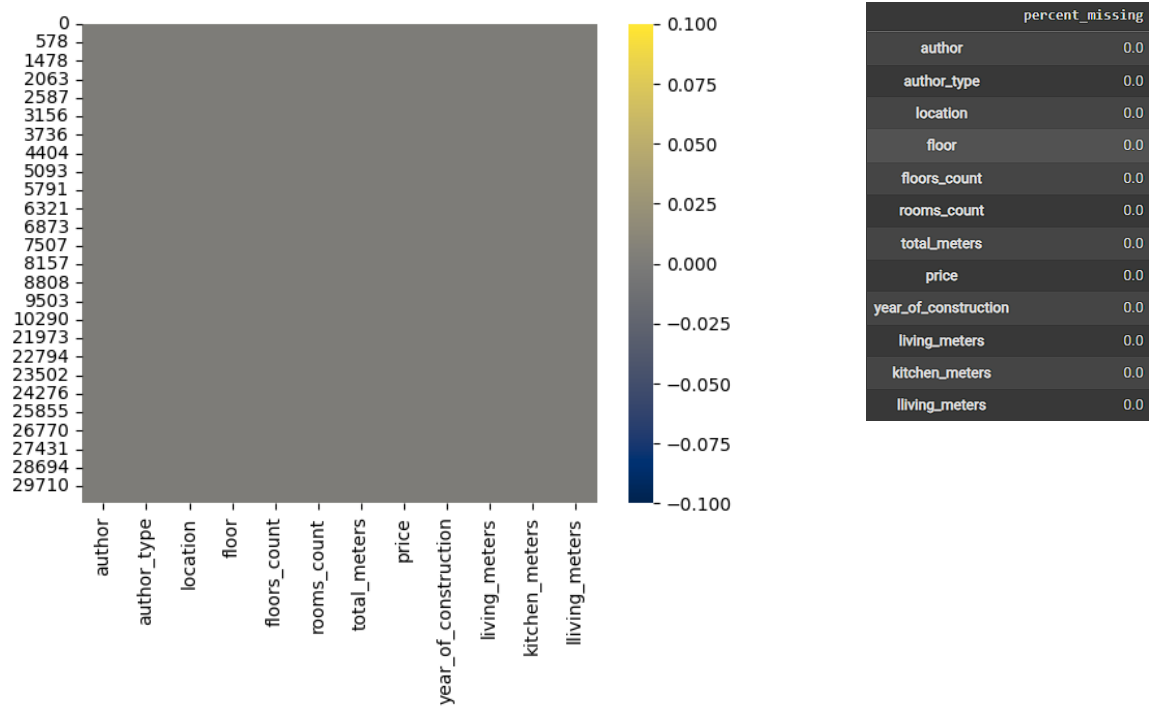
Удаление строк с пустыми значениями:

```
[ ] df.dropna(how='all', inplace=True)
df.dropna(subset=['price', 'location', 'author', 'author_type', 'floors_count',
                 'rooms_count', 'living_meters', 'kitchen_meters', 'year_of_construction'], inplace=True)
```

Проверка результата очистки:



До обработки




После обработки

Классификация данных:

```
[ ] df['year_of_construction'] = df['year_of_construction'].astype(int)
df['floor'] = df['floor'].astype(int)
df['floors_count'] = df['floors_count'].astype(int)
df['rooms_count'] = df['rooms_count'].astype(int)
df['price'] = df['price'].astype(int)

df.info()
```

 <class 'pandas.core.frame.DataFrame'>
Index: 8088 entries, 0 to 30652
Data columns (total 12 columns):

| # | Column | Non-Null Count | Dtype |
|----|----------------------|----------------|---------|
| 0 | author | 8088 non-null | object |
| 1 | author_type | 8088 non-null | object |
| 2 | location | 8088 non-null | object |
| 3 | floor | 8088 non-null | int64 |
| 4 | floors_count | 8088 non-null | int64 |
| 5 | rooms_count | 8088 non-null | int64 |
| 6 | total_meters | 8088 non-null | float64 |
| 7 | price | 8088 non-null | int64 |
| 8 | year_of_construction | 8088 non-null | int64 |
| 9 | living_meters | 8088 non-null | float64 |
| 10 | kitchen_meters | 8088 non-null | float64 |
| 11 | lliving_meters | 8088 non-null | float64 |

dtypes: float64(4), int64(5), object(3)
memory usage: 821.4+ KB

Очистка данных от выбросов:

```
[ ] q1 = df['price'].quantile(0.25)
q3 = df['price'].quantile(0.75)
iqr = q3 - q1
df
df = df[(df['price'] < q3 + 1.5 * iqr) & (df['price'] > q1 - 1.5 * iqr)].reset_index()

df.drop(['index'], axis = 1, inplace = True)
```

Для очистки данных от выбросов, я использовал «Метод IQR»

3. Анализ данных

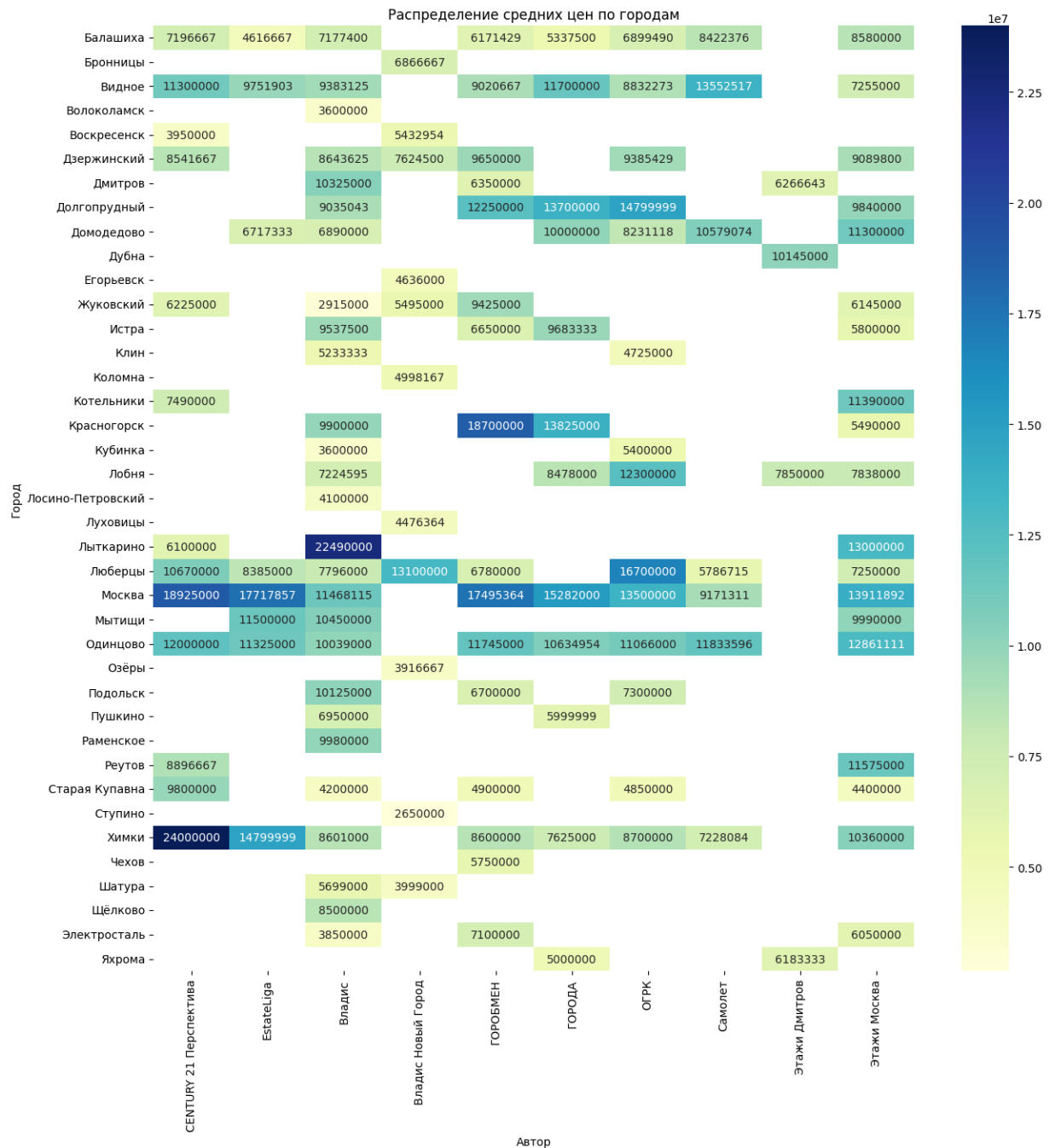
Первичный (общий) анализ данных:



Я сделал круговые диаграммы с топ 5 по популярности данными по четырем колонкам (город, автор, год постройки здания и тип автора).

На первой круговой диаграмме «Популярность городов» можно увидеть, что большинство квартир (объявлений) находятся в городе Москва – 37.0%. На второй «Популярность авторов объявлений» самым популярным оказалось агентство «Владис» - 38.8%. На третьей диаграмме «Популярность по году постройки» лидирующим годом оказался 2024 – 26.5%. А на четвертой «Популярность по типу автора» самым популярным стал real_estate_agent (агент по недвижимости) – 49.6%.

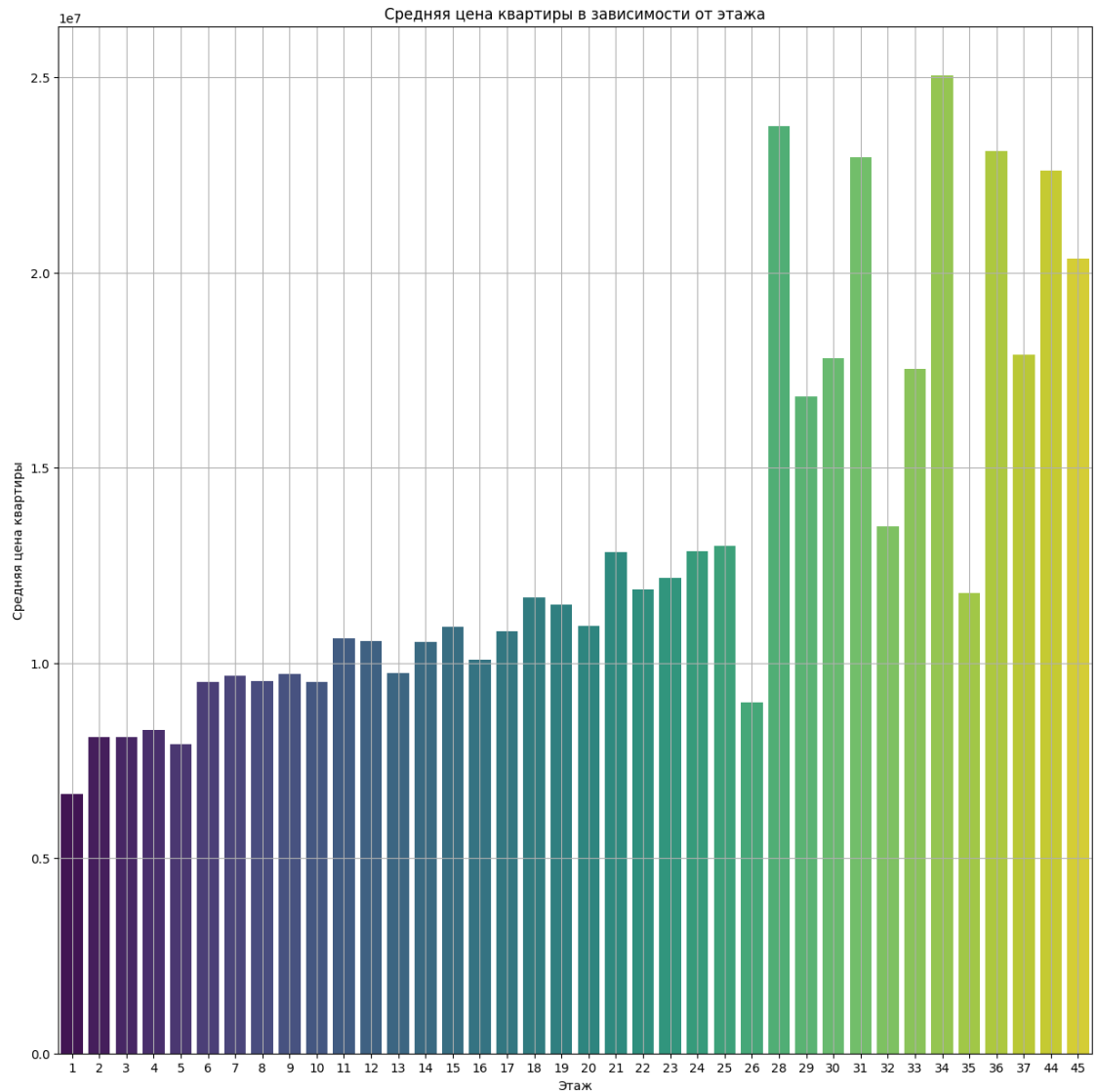
Тепловая карта отношения городов к авторам по средней цене:



Я сделал тепловую карту с отношением городов к 10 популярным авторам по средней цене.

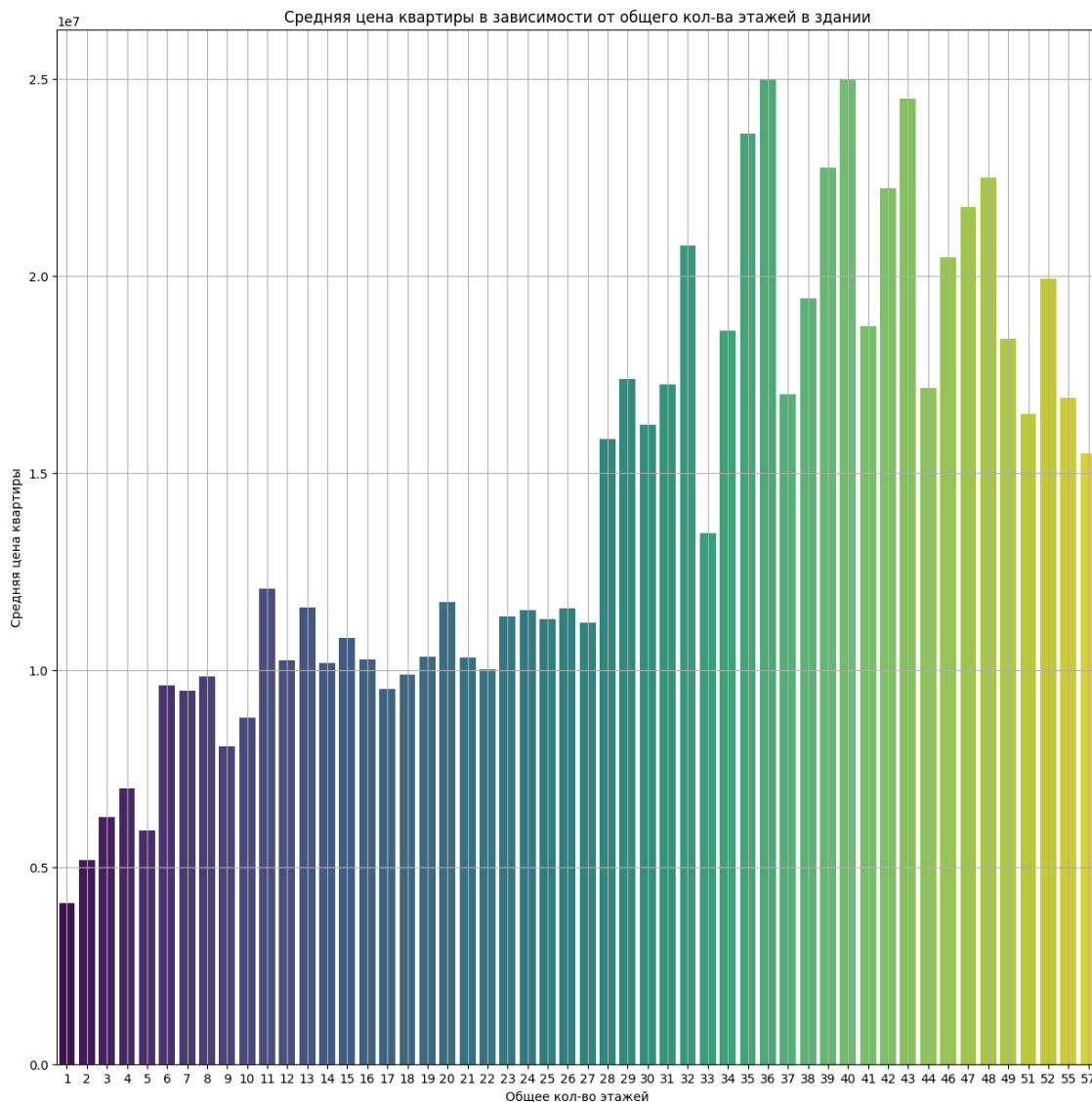
По ней можно отметить, что самые дорогие средние цены по 10 популярным авторам у городов: Москва, Одинцово, Мытищи, Красногорск и Химки. А самые дешевые: Кубинка, Шатура. Самыми популярными городами среди авторов объявлений оказались: Москва, Одинцово, Люберцы, Химки, Балашиха и Видное. А самыми непопулярными: Ступино, Щёлково, Чехов, Озёры, Луховицы, Коломна, Егорьевск.

Зависимость средней цены квартиры от этажа:



Я построил график зависимости средних цен от этажей.

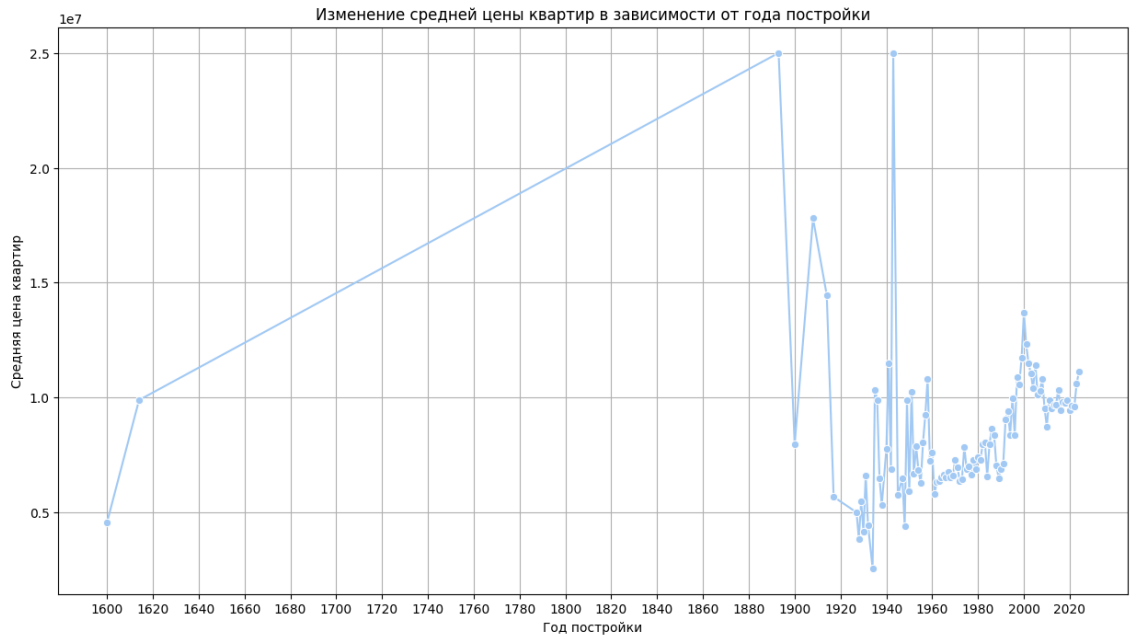
На графике видно, что зависимость присутствует: чем выше этаж — тем дороже квартира. Но всё же график выглядит нестабильно и можно предположить, что цена больше зависит от общего кол-ва этажей, а не этажа квартиры, проверим это:



Я построил график зависимости средних цен от общего кол-ва этажей здания.

Видно, что «График зависимости средних цен от общего кол-ва этажей здания» стабильнее чем «График зависимости средних цен от этажей», а значит моё предположение было верным: «Цена квартиры зависит от общего кол-ва этажей».

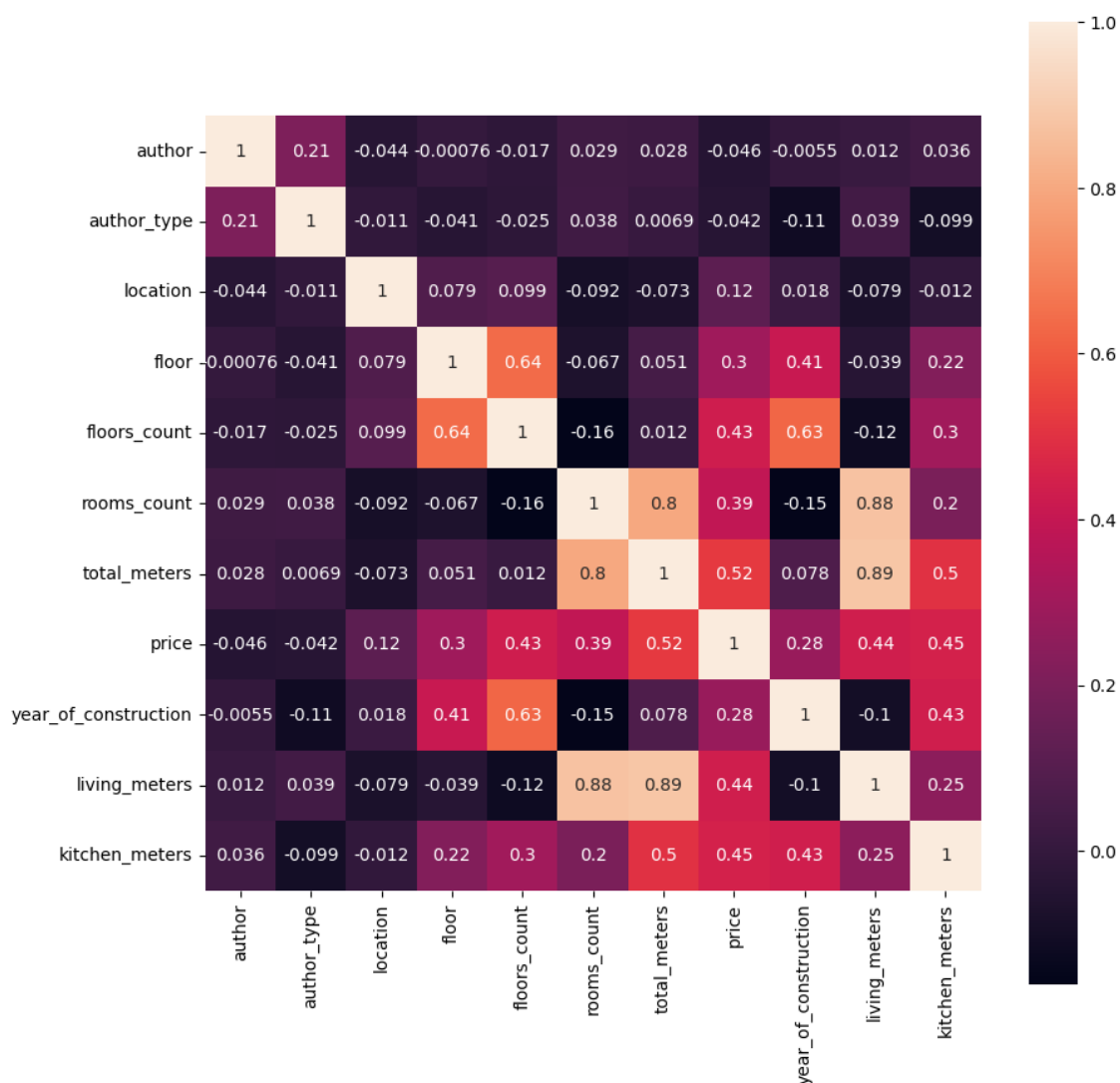
Зависимость цены квартиры от года постройки здания:



Я построил линейный график, который показывает зависимость цены от года постройки здания.

По графику можно увидеть, что очень мало объявлений со старыми зданиями. И можно увидеть, что год постройки здания влияет на цену квартиры: чем новее здание — тем дороже стоит квартира в нем.

Матрица корреляций:



Проанализировав Матрицу Корреляций можно увидеть, что с целевой переменной (ценой) есть зависимости:

Цена и локация – 0.12;

Цена и Этаж – 0.3;

Цена и кол-во этажей – 0.43;

Цена и кол-во комнат – 0.39;

Цена и общая площадь – 0.52;

Цена и год постройки здания – 0.28;

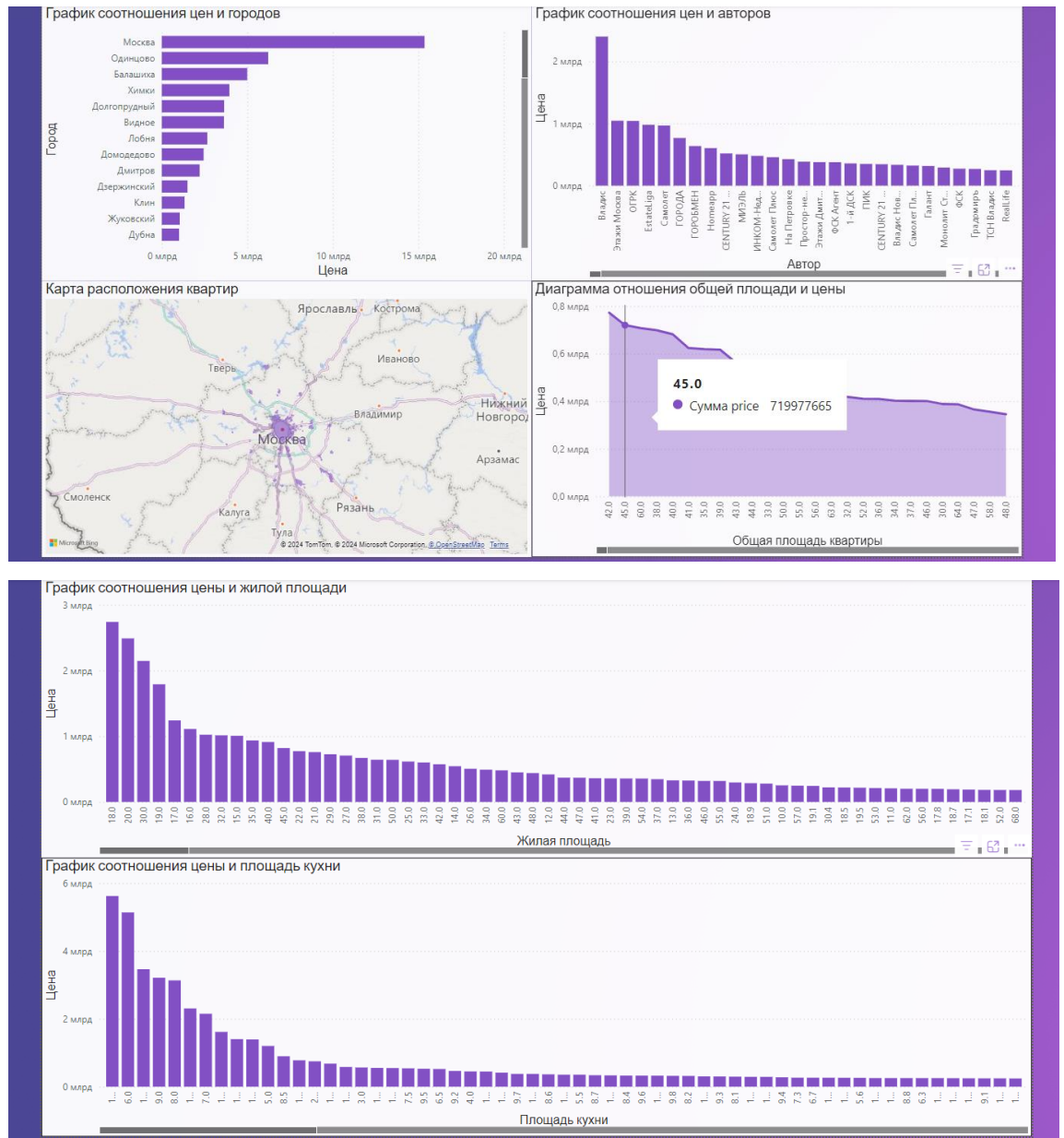
Цена и жилая площадь – 0.44;

Цена и площадь кухни – 0.45.

Также можно заметить связи между: этажом квартиры и общим кол-вом этажей здания; общей площадью и кол-вом комнат; годом постройки здания и

кол-вом этажей в здании; жилой площадью и кол-вом комнат; жилой площадью и общей площадью.

Анализ с помощью Power BI:



С помощью Power BI я вывел зависимости суммы цены от различных факторов и карту расположения квартир.

В первых двух графиках лидируют Москва и агентство «Владис» из-за их большой популярности среди объявлений. По диаграмме и оставшимся двум графикам можно сказать, что чем больше площади — тем выше цена квартиры.

Заключение

После очистки данных и проведение анализа с помощью различных инструментов можно сказать, что зависимостей с целевой переменной (ценой) достаточно много: локация, этаж, кол-во этажей, кол-во комнат, общая площадь, год постройки здания, жилая площадь, площадь кухни. Но самое большое влияние на цену оказывают: кол-во этажей, кол-во комнат, общая площадь, жилая площадь, площадь кухни.

По анализу других данных можно увидеть зависимости: этажом квартиры и общим кол-вом этажей здания, общей площадью и кол-вом комнат, годом постройки здания и кол-вом этажей в здании, жилой площадью и кол-вом комнат, жилой площадью и общей площадью.

Матрица корреляций сыграла ключевую роль в анализе, так как по ней видны все зависимости, а остальные построенные графики хорошо визуально отображали зависимости по целевой переменной.

Недочеты: я мог бы построить больше графиков и регрессии, лучше очистить данные от выбросов и построить модель.