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

Исследовательский аналитический анализ:

«Модель оценки цены квартиры на рынке по Московскому региону: Москва, Новая Москва, Московская область»

Работу выполнила:
Студентка группы № ИСП-21
Семрягина Анна
Проверил:
преподаватель
Базяк Г. В.

Долгопрудный, 2024

АННОТАЦИЯ

В современных условиях становится особенно важным отслеживать изменения стоимости жилья и выбирать удачный момент для покупки недвижимости в Москве и Московской области. Постоянный мониторинг цен и учет рыночных трендов позволяют не только выбрать наиболее подходящее время для приобретения, но и оценить факторы, которые могут повлиять на стоимость в будущем — такие как изменения в инфраструктуре, новые застройки, транспортная доступность и экологическая ситуация в районах. Грамотный подход к анализу рынка помогает минимизировать риски, оптимизировать затраты и принять более взвешенное решение, что особенно актуально в условиях нестабильной экономической обстановки.

Целью анализа является: собрать данные и провести разведочный исследовательский анализ данных (EDA) для построения модели, которая будет оценивать цену квадратного метра недвижимости в Московском регионе (Москва, Новая Москва, Московская область).

Задачи, которые нужно выполнить для достижения цели:

1. составить список параметров, значительно влияющих на цену квадратного метра жилой площади;
2. с учётом выявленных выше факторов произвести парсинг данных по квартирам на продажу, используя различные парсеры;
3. произвести подготовку данных для анализа: проверка на пропуски, выбросы и ошибки;
4. обработать выявленные аномалии (удалить / заполнить)

ВВЕДЕНИЕ

Рынок жилой недвижимости Москвы крайне динамичен, и его изучение позволяет определять наиболее подходящее время для приобретения жилья, что может принести значительную экономию средств для потенциального покупателя. Москва, как столица Российской Федерации, привлекает большое число желающих приобрести недвижимость, из-за чего спрос на жилье здесь стабильно высок. Однако на рынке наблюдаются определенные закономерности, когда стоимость жилья снижается, открывая выгодные возможности для приобретения.

Кроме того, рынок столичной недвижимости чувствителен к внешним и внутренним экономическим факторам, таким как изменения в ипотечных ставках, политическая и экономическая ситуация, а также государственные меры регулирования. Эти факторы могут значительно влиять на колебания цен и создавать как благоприятные, так и неблагоприятные условия для покупателей. Исследование этих факторов и их влияние на стоимость жилья позволяет глубже понимать, какие изменения могут произойти на рынке в краткосрочной и долгосрочной перспективе.

Сначала составим перечень параметров, влияющих на стоимость квадратного метра: местоположение, тип здания, этажность и планировка квартиры, а также её состояние на момент продажи. Определив ключевые параметры, перейдем к парсингу данных.

```python
import cianparser
from time import sleep

a = 0
while a < 54:
    moscow_parser = cianparser.CianParser(location="Москва")
    data = moscow_parser.get_flats(
        deal_type="sale",
        rooms=("all"),
        with_saving_csv=True,
        additional_settings={"start_page": 1 + a, "end_page": 2 + a},
        with_extra_data=True,
    )
    sleep(60)
    a += 2
```

Просмотрим полученный набор данных:

```
df = pd.read_csv("intensity.csv")
df.head()
```

	author	author_type	url	location	deal_type	accommodation_type	floor	floors_count	rooms_count	total_meters	...	heating_type	finish_type	living_meters	kitchen_meters	phone	district	street	house_number	underground	residential_complex
0	Метражи group	real_estate_agent	https://www.cian.ru/sale/flat/308167237/	Москва	sale	flat	5	7	1	34.6	...	NaN	NaN	18.0	8.0	7.985041e+10	Северное Измайлово	15-я Парковая	54	Щёлковская	NaN
1	ЛэндЛ	real_estate_agent	https://www.cian.ru/sale/flat/302263383/	Москва	sale	flat	14	45	1	41.3	...	NaN	Без отделки	20.0	11.0	7.964559e+10	Нижегородский	Перовское шоссе	NaN	Нижегородская	Level Нижегородская
2	Stenoy	developer	https://www.cian.ru/sale/flat/300878920/	Москва	sale	flat	10	12	1	34.4	...	NaN	Без отделки, черновая, чистовая	11.8	11.7	7.497716e+10	Преображенское	Электрозаводская	60	Преображенская площадь	ARTEL
3	Alliance Agency Real Estate	real_estate_agent	https://www.cian.ru/sale/flat/298254403/	Москва	sale	flat	4	33	1	42.9	...	NaN	NaN	22.9	15.0	7.965188e+10	Останкинский	Годовикова	11к2	Алексеевская	iLove
4	Зиля Карамова	realtor	https://www.cian.ru/sale/flat/263316279/	Москва	sale	flat	1	16	1	37.7	...	NaN	NaN	NaN	NaN	7.916094e+10	Чертаново Центральное	Варшавское шоссе	142К2	Пражская	NaN

5 rows × 24 columns

Для исследования были собраны данные из открытых источников — с сайта недвижимости ЦИАН с помощью библиотеки cianparser. Итоговый набор данных включает 11 тысяч записей.

Первичный анализ показал, что использование исходных данных для оценки стоимости жилья было затруднено из-за наличия значительных выбросов. Поэтому для подготовки данных к анализу я провела очистку от пустых значений и выделила ключевые показатели, такие как минимальная и максимальная площадь квартиры и кухни, а также средняя цена объекта по Москве и области. В выборку вошли объекты как первичного, так и вторичного рынка. Для выполнения задачи подготовки данных я использовала несколько библиотек, включая pandas, numpy и другие.

Поскольку работа с библиотеками мне менее знакома, я применяла два метода очистки данных: с помощью Excel и с использованием функции drop.

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10509 entries, 0 to 10508
Data columns (total 24 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   author                10339 non-null  object
 1   author_type           10336 non-null  object
 2   url                   10506 non-null  object
 3   location              10102 non-null  object
 4   deal_type             10509 non-null  object
 5   accommodation_type    10509 non-null  object
 6   floor                 10509 non-null  int64
 7   floors_count          10509 non-null  int64
 8   rooms_count           10509 non-null  int64
 9   total_meters          10509 non-null  float64
 10  price                 10476 non-null  float64
 11  year_of_construction  8313 non-null   object
 12  object_type           0 non-null      float64
 13  house_material_type   1242 non-null   object
 14  heating_type          0 non-null      float64
 15  finish_type           1702 non-null   object
 16  living_meters         8085 non-null   float64
 17  kitchen_meters        8996 non-null   float64
 18  phone                 10506 non-null  float64
 19  district              5934 non-null   object
...
 22  underground           6628 non-null   object
 23  residential_complex   4765 non-null   object
dtypes: float64(7), int64(3), object(14)
memory usage: 1.9+ MB
```

После получения информации, стоит избавится от значений «-1» и заменить в столбцах floor, floors_count и rooms_count тип данных с float на int.

```
# Здесь мы заменяем все значения -1 и -1.0 на NaN, для удобной работы с данными
df = df.replace(-1,np.nan)
df = df.replace("-1",np.nan)
df = df.replace(-1.0,np.nan)
df = df.replace("-1.0",np.nan)
# Удаляем строки, в которых есть пустые значения, т.к. они влияют на цену
df = df.dropna(subset=['location', 'rooms_count', 'price'])
# Переводим float значения в столбцах floor; floors_count; rooms_count,
#  т.к. они не могут иметь в себе например 2.5 комнаты
df['floor'] = df['floor'].astype(int)
df['floors_count'] = df['floors_count'].astype(int)
df['rooms_count'] = df['rooms_count'].astype(int)
# Удаляем ненужные столбцы, которые не несут нужной информации которая может повлиять на образование цена квартиры,
# или данные которые имеют только одно значение, такие как deal_type; accomodation_type - они имеют одно статичное значение,
#  тип сделки: продажа, тип помещения - квартира
df.drop(['phone', 'deal_type', 'accommodation_type', 'object_type', 'heating_type'], axis=1, inplace=True)
print(f'Количество столбцов после чистки {df.shape[1]} столбцы')
# Также необходимо в сохранённом файле изменить ��� на 0, для избежания ошибок
```

Выводим данные после чистки:

```
df.isnull().sum()
```

author	146
author_type	146
url	0
location	0
floor	0
floors_count	0
rooms_count	0
total_meters	0
price	0
year_of_construction	2099
house_material_type	8871
finish_type	8458
living_meters	2316
kitchen_meters	1407
district	4476
street	1393
house_number	1037
underground	3489
residential_complex	5473

dtype: int64

Далее проверка на процентный пропуск в данных:

```
for col in df.columns:
    pct_missing = np.mean(df[col].isnull())
    print('{} - {}%'.format(col, round(pct_missing*100)))
```

```
author - 1%
author_type - 1%
url - 0%
location - 0%
floor - 0%
floors_count - 0%
rooms_count - 0%
total_meters - 0%
price - 0%
year_of_construction - 21%
house_material_type - 88%
finish_type - 84%
living_meters - 23%
kitchen_meters - 14%
district - 44%
street - 14%
house_number - 10%
underground - 35%
residential_complex - 54%
```

В столбцах house_material_type и finish_type пропущенных значений более 50%, то принято решение удалить их.

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

Снова запускаем проверку на процентные пропуски:

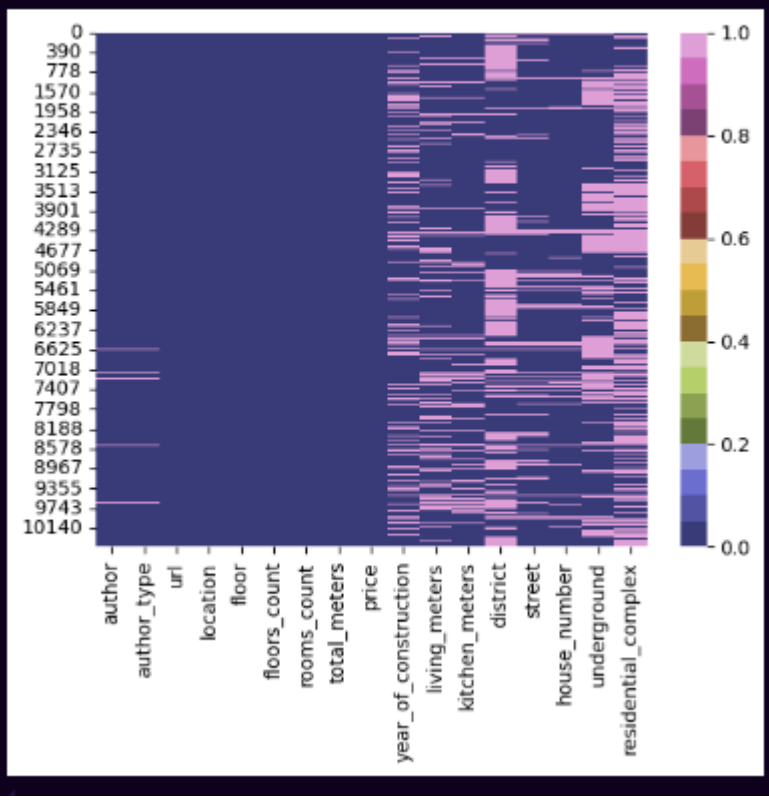
```
for col in df.columns:
    pct_missing = np.mean(df[col].isnull())
    print('{} - {}%'.format(col, round(pct_missing*100)))
```

```
author - 1%
author_type - 1%
url - 0%
location - 0%
floor - 0%
floors_count - 0%
rooms_count - 0%
total_meters - 0%
price - 0%
year_of_construction - 21%
living_meters - 23%
kitchen_meters - 14%
district - 44%
street - 14%
house_number - 10%
underground - 35%
residential_complex - 54%
```

Сделаем хитмап для визуального представления пропусков в столбцах:

```
sns.heatmap(df.isnull(),cmap='tab20b')
```

```
<Axes: >
```



Мы заполняем пропуски (NaN, что означает "Не-число") наиболее часто встречающимся значением – модой и проверяем процент пропусков

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

```
for col in df.columns:
    pct_missing = np.mean(df[col].isnull())
    print('{} - {}%'.format(col, round(pct_missing*100)))
```

```
author - 1%
author_type - 1%
url - 0%
location - 0%
floor - 0%
floors_count - 0%
rooms_count - 0%
total_meters - 0%
price - 0%
year_of_construction - 21%
living_meters - 23%
kitchen_meters - 14%
district - 44%
street - 14%
house_number - 10%
underground - 35%
residential_complex - 54%
```

Так как наши собранные данные не везде имеют в записях район, и процент пропусков 35%, принято решение заполнить недостающие данные в колонке значениями из колонки с локацией объявления

```python
# Функция для замены пропусков в колонке 'district'
def fill_district(row):
    if pd.isna(row['district']):
        if row['location'] == 'Москва':
            if pd.notna(row['underground']):
                return row['underground']
            else:
                return 'Москва'
        else:
            return row['location']
    else:
        return row['district']
df['district'] = df.apply(fill_district, axis=1)
```

```python
for col in df.columns:
    pct_missing = np.mean(df[col].isnull())
    print('{} - {}%'.format(col, round(pct_missing*100)))
```

```
author - 1%
author_type - 1%
url - 0%
location - 0%
floor - 0%
floors_count - 0%
rooms_count - 0%
total_meters - 0%
price - 0%
year_of_construction - 21%
living_meters - 23%
kitchen_meters - 14%
district - 0%
street - 14%
house_number - 10%
underground - 35%
residential_complex - 54%
```

Затем уберём столбцы house_number и residential_complex, так как вновь большие пропуски в значениях, и влияния на цену они не оказывают

```python
df.drop(['residential_complex', 'house_number'], axis=1, inplace=True)
df = df.dropna(subset=['underground', 'street'])
```

Теперь посмотрим сколько пропущенных значений осталось в итоге:

```
for col in df.columns:
    pct_missing = np.mean(df[col].isnull())
    print('{} - {}%'.format(col, round(pct_missing*100)))
```

```
author - 1%
author_type - 1%
url - 0%
location - 0%
floor - 0%
floors_count - 0%
rooms_count - 0%
total_meters - 0%
price - 0%
year_of_construction - 18%
living_meters - 24%
kitchen_meters - 13%
district - 0%
street - 0%
underground - 0%
```

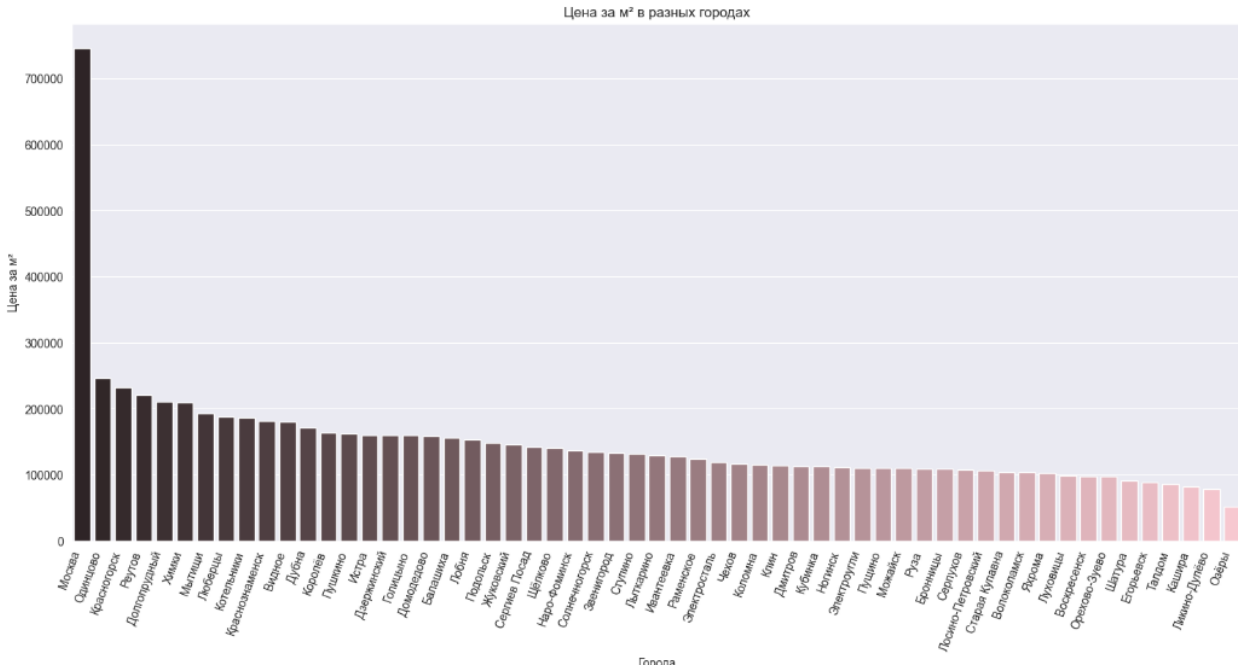
Пора добавить новых значений, поэтому пропишем новую функцию для вычисления средней цены за кв. метр в каждом городе

```
list_city = df['location'].unique()

def price_for_meter(location):
    city = df[df['location'] == location]
    price_for_city = city['price'].sum()

    clean_data = city['total_meters'].sum()

    return round(price_for_city/clean_data, 2)

with open('info_of_city_and_price.csv', 'w', newline='', encoding='UTF-8') as csvfile:
    names = ['city', 'price_for_meter']
    writer = csv.DictWriter(csvfile, fieldnames=names)
    writer.writeheader()
    for city in list_city:
        writer.writerow({'city': city, 'price_for_meter': price_for_meter(city)})
```

Сортируем данные и создаем график:

```
info = pd.read_csv('sorted_info_of_city_and_price.csv')
info_sorted = info.sort_values(by='price_for_meter', ascending=False)
```

```
# График для средней цены за м2 во всех городах
sns.set_style("darkgrid")
info = pd.read_csv('sorted_info_of_city_and_price.csv')
plt.figure(figsize=(18, 8))
sns.barplot(hue='city', legend=False, x='city', y='price_for_meter', data=info, color='pink')
plt.title('Цена за м² в разных городах')
plt.xlabel('Города')
plt.ylabel('Цена за м²')
plt.xticks(rotation=70, ha='right')
plt.show()
```

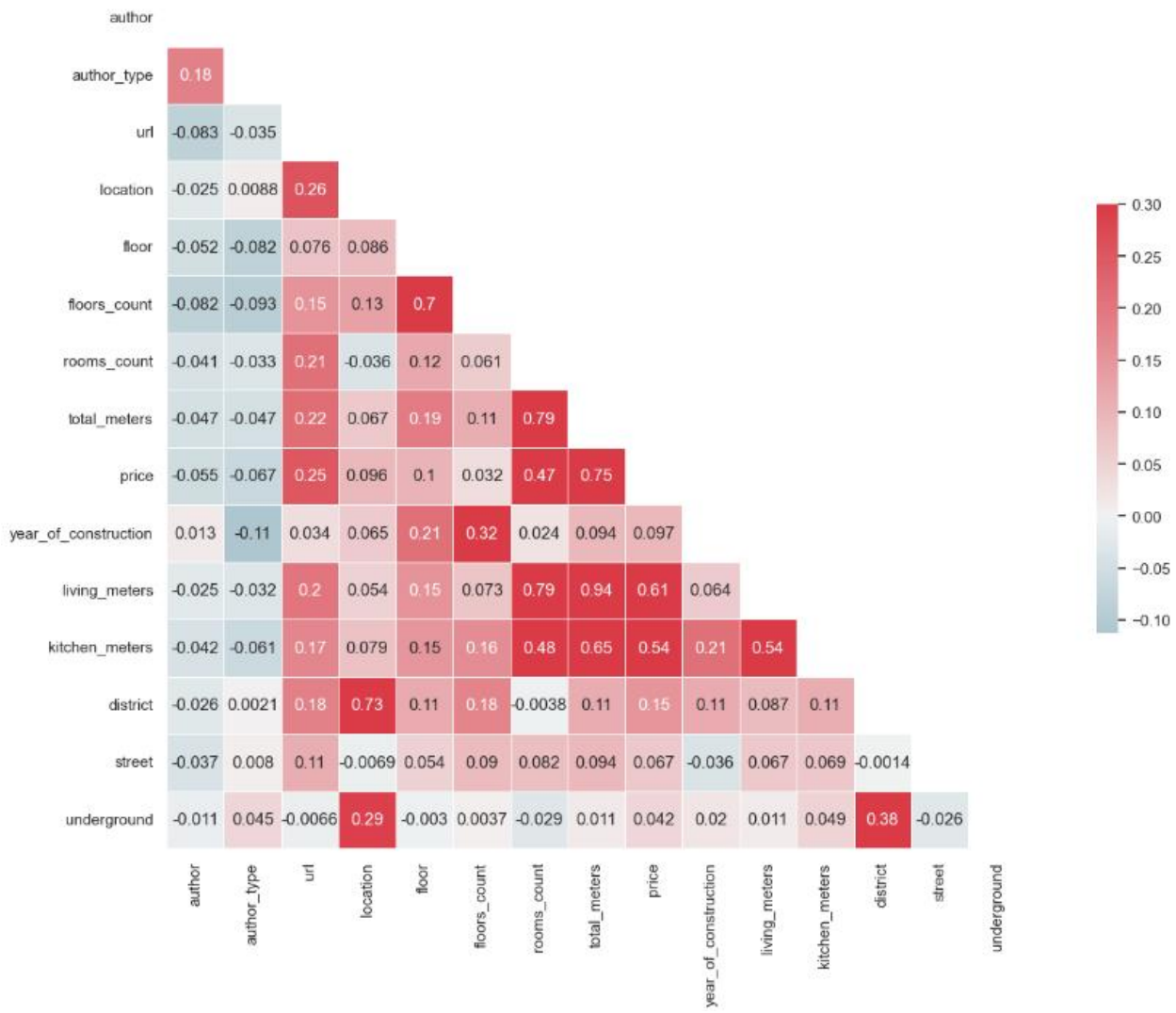


Перекодируем значения в строках если они имеют тип – строка

```
from sklearn import preprocessing
#функция, которая принимает на вход наши данные, кодирует числовыми значениями категориальные признаки
#и возвращает обновленный данные и сами кодировщики
def number_encode_features(init_df):
    result = init_df.copy() #копируем нашу исходную таблицу
    encoders = {}
    for column in result.columns:
        if result.dtypes[column] == object: # np.object -- строковый тип / если тип столбца - строка, то нужно его закодировать
            encoders[column] = preprocessing.LabelEncoder() #для колонки column создаем кодировщик
            result[column] = encoders[column].fit_transform(result[column]) #применяем кодировщик к столбцу и перезаписываем столбец
    return result, encoders

encoded_data, encoders = number_encode_features(df) #Теперь encoded data содержит закодированные категориальные признаки
encoded_data.head() #проверяем
```

Создаем визуализацию матрицы корреляции с помощью хитмапа, который помогает быстро оценить взаимосвязи между переменными в наборе данных

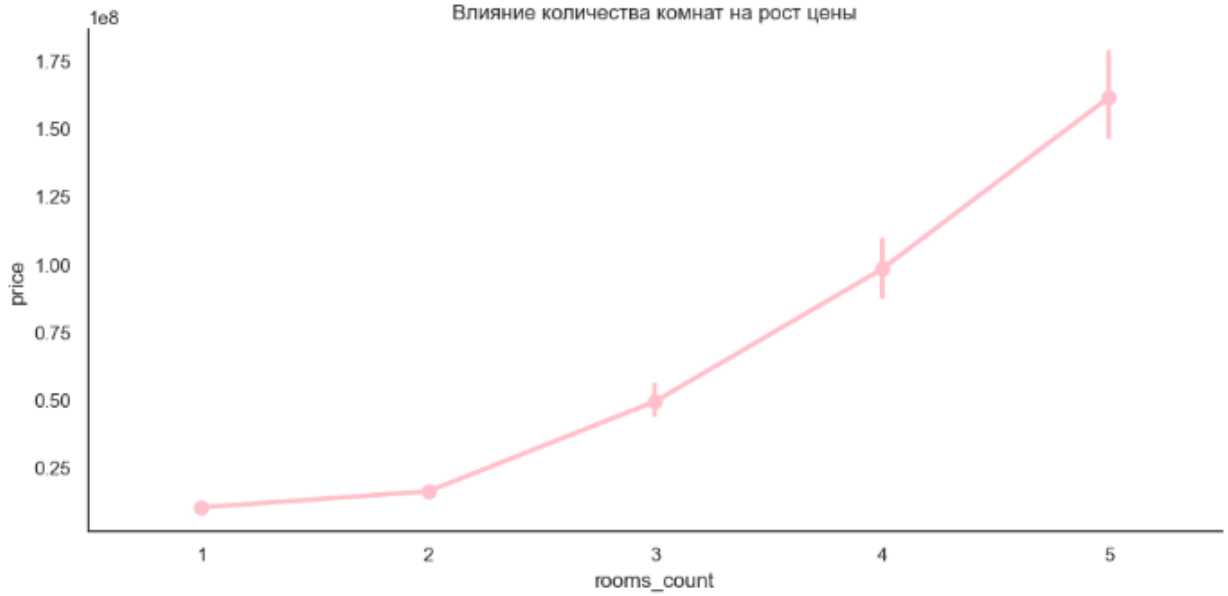
```
sns.set(style="white")


temp3 = encoded_data.copy()
corr = temp3.corr()
mask = np.zeros_like(corr, dtype=bool)
mask[np.triu_indices_from(mask)] = True
f, ax = plt.subplots(figsize=(18, 11))
cmap = sns.diverging_palette(220, 10, as_cmap=True)
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0, annot=True,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
```



Заметим взаимосвязь между ценой, количеством комнат и общим метражом квартиры.

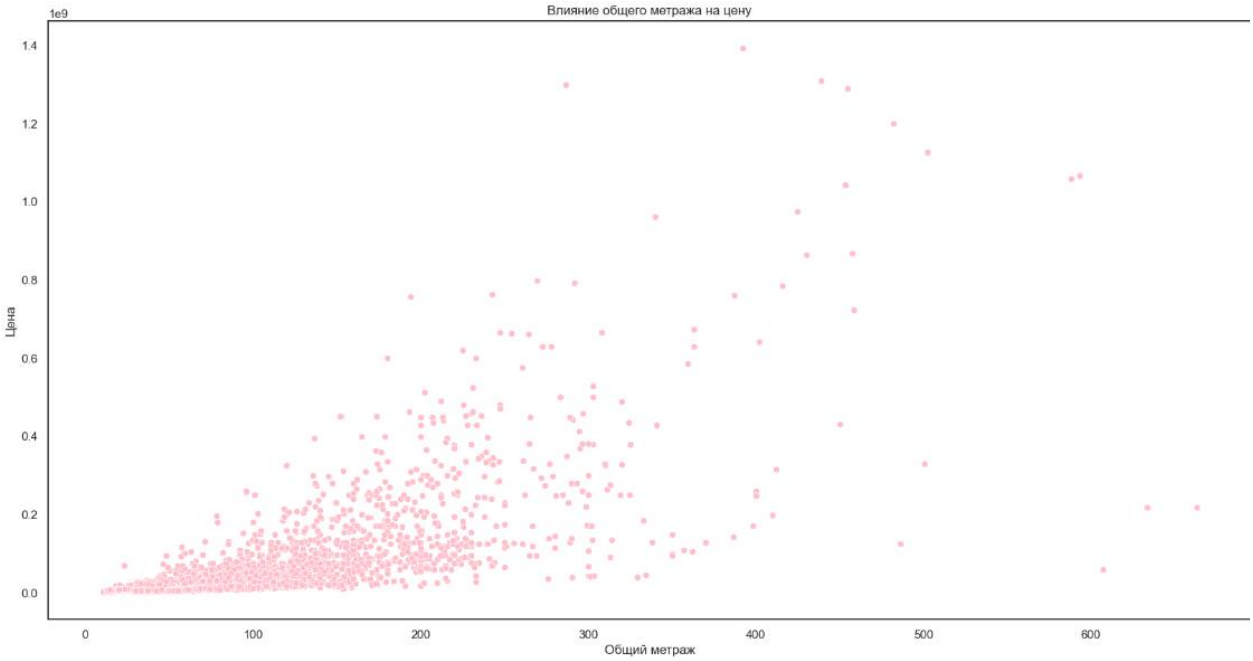
Теперь составим график влияния количества комнат на цену жилья:

```
plt.figure()
sns.catplot(x='rooms_count', y='price', data=encoded_data, kind='point', aspect=2, color='pink')
plt.title("Влияние количества комнат на рост цены")
```



Стоит немного разбавить графики, диаграммой поэтому сделаем диаграмму рассеяния, в которой показано отношение цены к общему метражу, взаимосвязь между которыми нашли раннее в корреляции

```
plt.figure(figsize=(20, 10))
sns.scatterplot(x='total_meters', y='price', data=encoded_data, color='pink')
plt.title('Влияние общего метража на цену')
plt.xlabel('Общий метраж')
plt.ylabel('Цена')
plt.show()
```



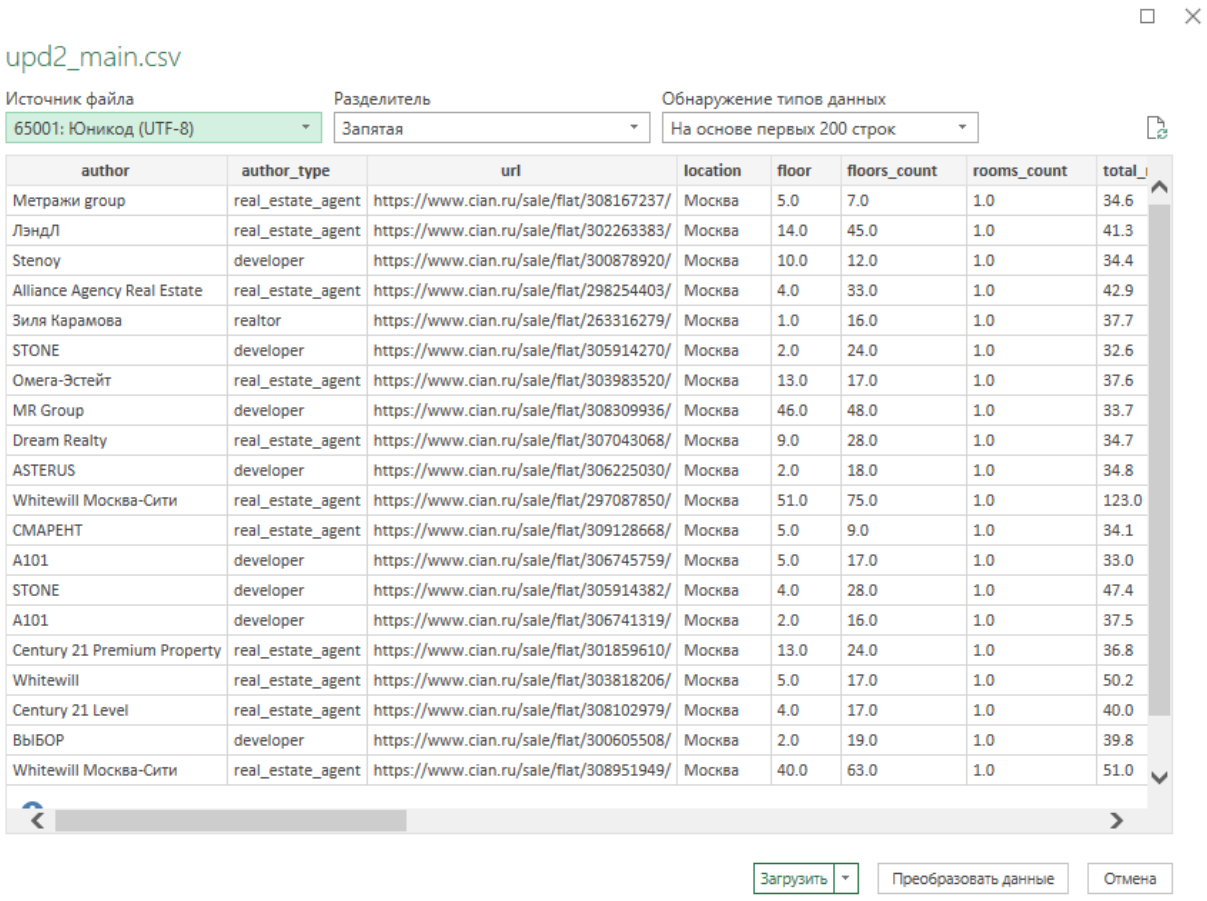
После вывода графика, сделаем вывод что чем больше метраж, тем выше цена, однако присутствуют критические значения, которые либо имеют

очень большую стоимость, либо наоборот очень дешёвые для своего метража.

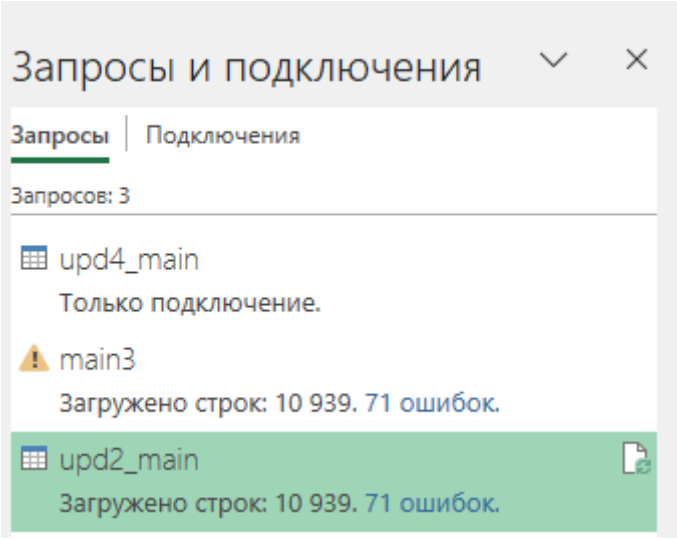
Весь программный код для работы с данными, написанный на языке Python, доступен по ссылке «https://github.com/aanchik/intensive_plane».

Теперь рассмотрим обработку данных в программе Excel:

Создаем новый файл и на вкладке Данные вытягиваем запрос на данные из собранного нами csv файла, встроенный редактор сам распознает кодировку и разделители поэтому просто загружаем данные



author	author_type	url	location	floor	floors_count	rooms_count	total_
Метражи group	real_estate_agent	https://www.cian.ru/sale/flat/308167237/	Москва	5.0	7.0	1.0	34.6
ЛэндЛ	real_estate_agent	https://www.cian.ru/sale/flat/302263383/	Москва	14.0	45.0	1.0	41.3
Stenoy	developer	https://www.cian.ru/sale/flat/300878920/	Москва	10.0	12.0	1.0	34.4
Alliance Agency Real Estate	real_estate_agent	https://www.cian.ru/sale/flat/298254403/	Москва	4.0	33.0	1.0	42.9
Зиля Карамова	realtor	https://www.cian.ru/sale/flat/263316279/	Москва	1.0	16.0	1.0	37.7
STONE	developer	https://www.cian.ru/sale/flat/305914270/	Москва	2.0	24.0	1.0	32.6
Омега-Эстейт	real_estate_agent	https://www.cian.ru/sale/flat/303983520/	Москва	13.0	17.0	1.0	37.6
MR Group	developer	https://www.cian.ru/sale/flat/308309936/	Москва	46.0	48.0	1.0	33.7
Dream Realty	real_estate_agent	https://www.cian.ru/sale/flat/307043068/	Москва	9.0	28.0	1.0	34.7
ASTERUS	developer	https://www.cian.ru/sale/flat/306225030/	Москва	2.0	18.0	1.0	34.8
Whitewill Москва-Сити	real_estate_agent	https://www.cian.ru/sale/flat/297087850/	Москва	51.0	75.0	1.0	123.0
СМАРЕНТ	real_estate_agent	https://www.cian.ru/sale/flat/309128668/	Москва	5.0	9.0	1.0	34.1
A101	developer	https://www.cian.ru/sale/flat/306745759/	Москва	5.0	17.0	1.0	33.0
STONE	developer	https://www.cian.ru/sale/flat/305914382/	Москва	4.0	28.0	1.0	47.4
A101	developer	https://www.cian.ru/sale/flat/306741319/	Москва	2.0	16.0	1.0	37.5
Century 21 Premium Property	real_estate_agent	https://www.cian.ru/sale/flat/301859610/	Москва	13.0	24.0	1.0	36.8
Whitewill	real_estate_agent	https://www.cian.ru/sale/flat/303818206/	Москва	5.0	17.0	1.0	50.2
Century 21 Level	real_estate_agent	https://www.cian.ru/sale/flat/308102979/	Москва	4.0	17.0	1.0	40.0
ВЫБОР	developer	https://www.cian.ru/sale/flat/300605508/	Москва	2.0	19.0	1.0	39.8
Whitewill Москва-Сити	real_estate_agent	https://www.cian.ru/sale/flat/308951949/	Москва	40.0	63.0	1.0	51.0



Встроенный отладчик сразу сообщает об ошибках в данных, которые в дальнейшем мы устраним.

Теперь начинаем работу с чисткой данных от пустых и критических значений, первоначально проверяем самый интересующий нас столбец – цена, чтобы это было быстро и удобно включаем фильтрацию на пустые строки, так как пустых строк в этом столбце оказалось не так много, заполняем их в ручную, то есть переходим по имеющейся ссылке объявления и копируем данные оттуда, приводим ячейки в общий формат.

F	G	H
total_meters	price	year_of_construction
50.2	34900000.0	2024
36.5	13000000.0	2024
59.35	101785250.0	2024
45.0	23274675.0	2024
43.09	20252300.0	2024

Было

F	G	H
otal_meters	price	year_of_construction
50.2	34900000	2024
36.5	13000000	2024
59.35	101785250	2024
45.0	23274675	2024
43.09	20252300	2024
45.3	27171760	2024
72.6	139410000	2024

Стало

Затем смотрим на столбец с датой сдачи здания в эксплуатацию, и убираем все что будет построено в 2025 г. и позже котлованы нам не нужны)), также с помощью фильтра выделяем эти значения и удаляем.

Теперь рассмотрим другие столбцы, которые тоже мало влияют, или вообще не имеют связи с изменением цены, в нашем случае это (улица, номер дома, номер телефона, автор и тип автора) поэтому удаляем столбцы.

Столбец ссылками я решила оставить, так как все строки имеют данные и по ним можно считать количество объявлений для дальнейшего анализа, однако никакой связи с ценой не присутствует.

Теперь перейдем к столбцу с общим метражом, и также с помощью фильтра выделим пустые значения и будем вынуждены их удалить, продублируем действия на столбцы с жилой площадью и площадью кухни.

Стоит обратить внимание на целочисленные значения,

C	D	E	F
floor	floors_count	rooms_count	total_meters
5.0	17.0	1.0	50.2
21.0	30.0	1.0	36.5
4.0	6.0	1.0	59.35
2.0	16.0	1.0	45.0
7.0	20.0	1.0	43.09
7.0	13.0	1.0	45.3
2.0	14.0	1.0	72.6

поэтому в столбце с указанием этажа, этажности дома и количества комнат меняем формат ячейки, получаем

C	D	E	F
floor	floors_count	rooms_count	total_meters
5	17	1	50.2
21	30	1	36.5
4	6	1	59.35
2	16	1	45.0
7	20	1	43.09
7	13	1	45.3
2	14	1	72.6

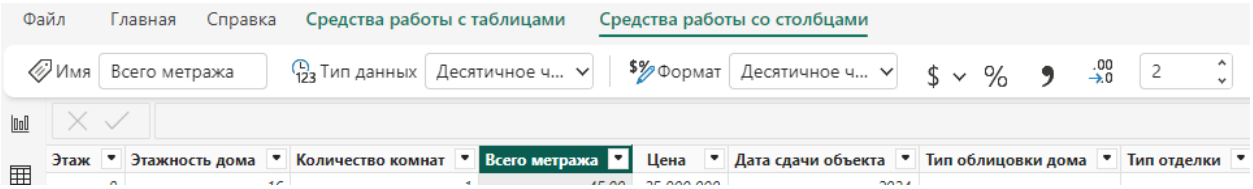
Общие метраж первоначально прописан дробными числами через ., это приведёт к ошибкам поэтому заменяем на ,

E	F	G
rooms_count	total_meters	price
1	50,2	34900000
1	36,5	13000000
1	59,35	101785250
1	45	23274675
1	43,09	20252300
1	45,3	27171760
1	72,6	139410000

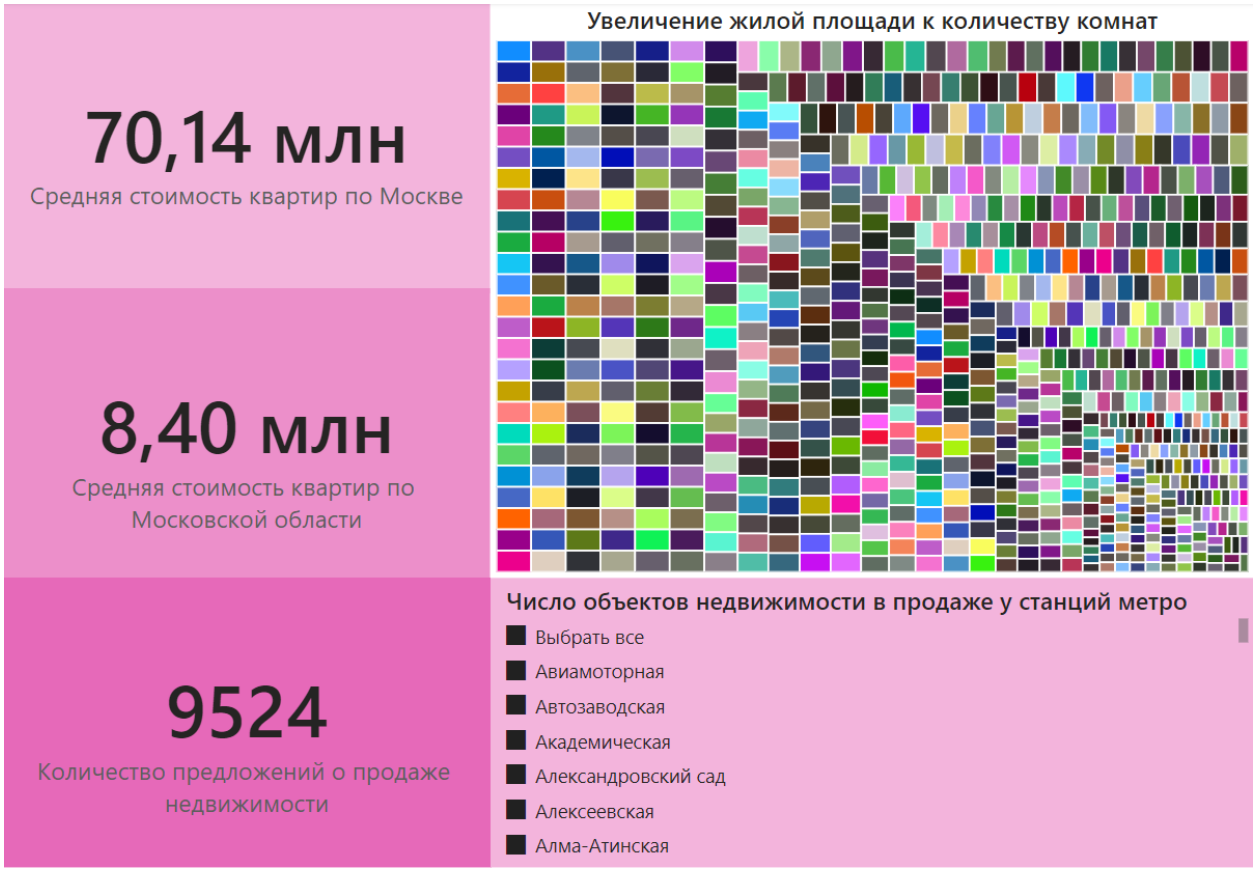
теперь данные готовы к анализу и визуализации.

ВИЗУАЛИЗАЦИЯ ДАННЫХ

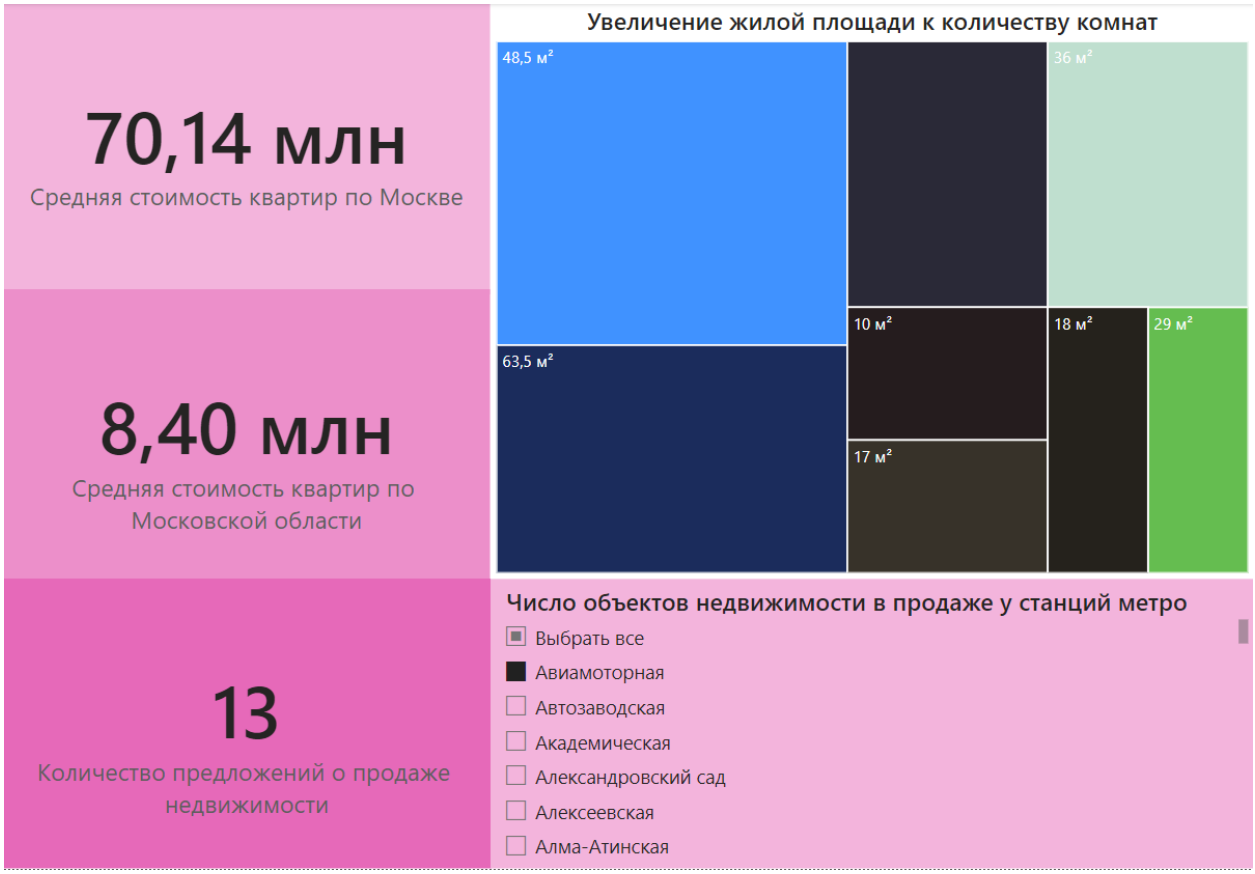
Визуализировать будем в программе Power BI, делаем запрос данных в наши csv файлы, тут есть два варианта как можно поступить, после обработки данных в Excel можно оставить формат данных .xlsx и работать с одним файлом, или же выбрать формат .csv однако если делить данные на Москву и область как это сделала я, то одним файлом не получится это сделать, и нужно отдельно сделать 3 запроса (общие данные, Москва, Московская область). Мне удобнее работать в формате .csv так как у нас большой набор данных и хранить их в одном файле большой риск.



Назначаем для каждого столбца тип данных, в целом это делается автоматически, но всегда стоит перепроверять, теперь данные готовы к работе и переходим к созданию дашборда.

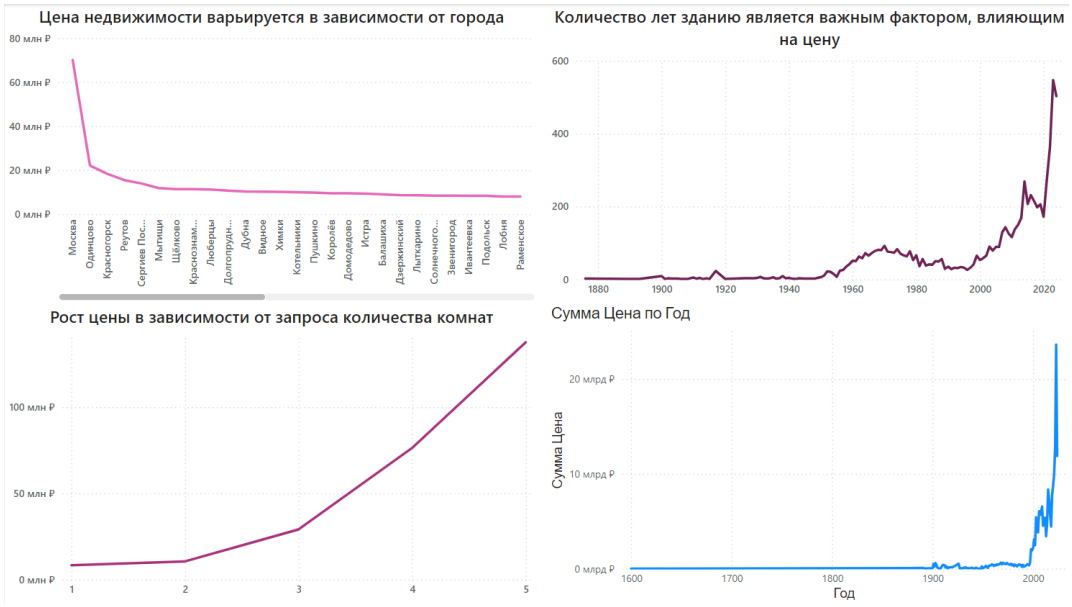


Первой дашборд представляет собой усредненные данные в показателях цены, и общее количество объявлений. Срез с указанием данных метро, представляет собой связь сколько предложений о продаже будет у той или иной станции. (актуально только для Москвы)



Пример наглядно показывает, что около станции Авиамоторная 13 предложений о продаже квартиры, и также изменился другой визуальный элемент, показывающий примерный размер площади к количеству комнат, на скриншоте отображены значения, которые напрямую связаны со станцией Авиамоторная.

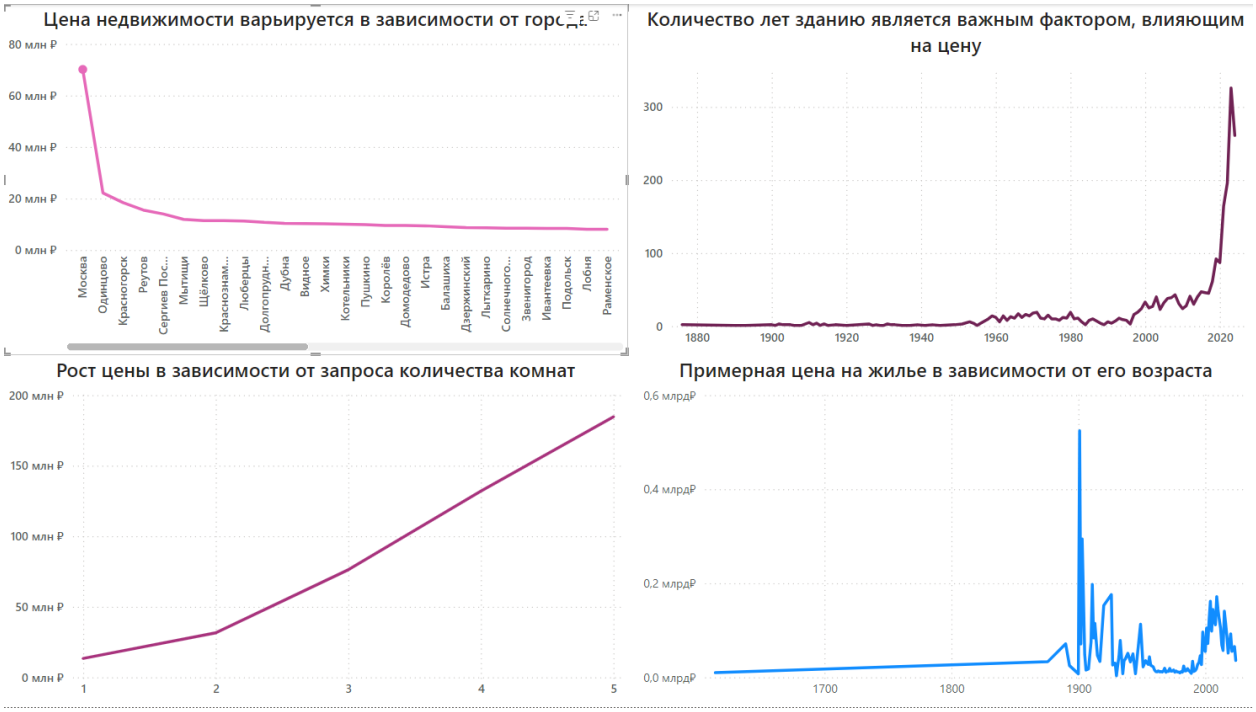
Теперь перейдем на второй дашборд, самый значимый для нас



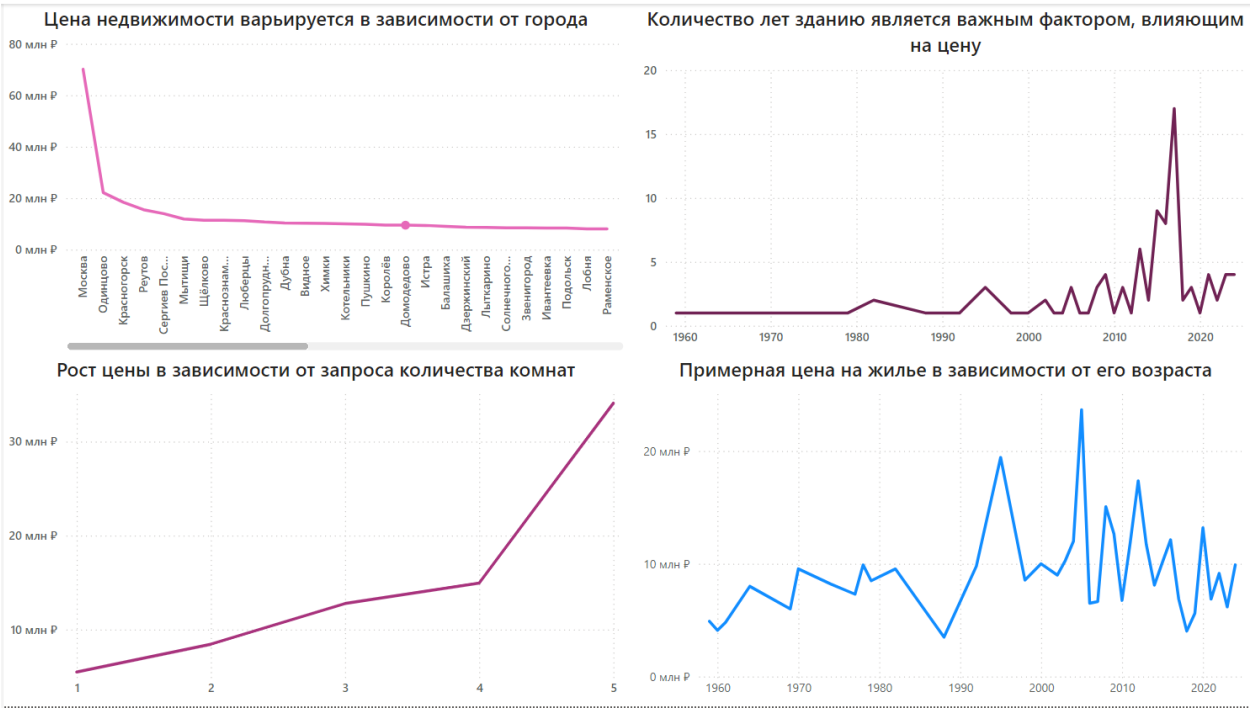
Здесь показаны 3 зависимости цены от ключевых параметров:

1. Расположение
2. Количество комнат
3. Дата сдачи здания в эксплуатацию

Рассмотрим каждый подробнее, первый график показал нам, что выше всего стоимость будет в Москве в отличие от других, потому что имеются объекты с очень большой ценой.



Показатели по Москве



Показатели города из области

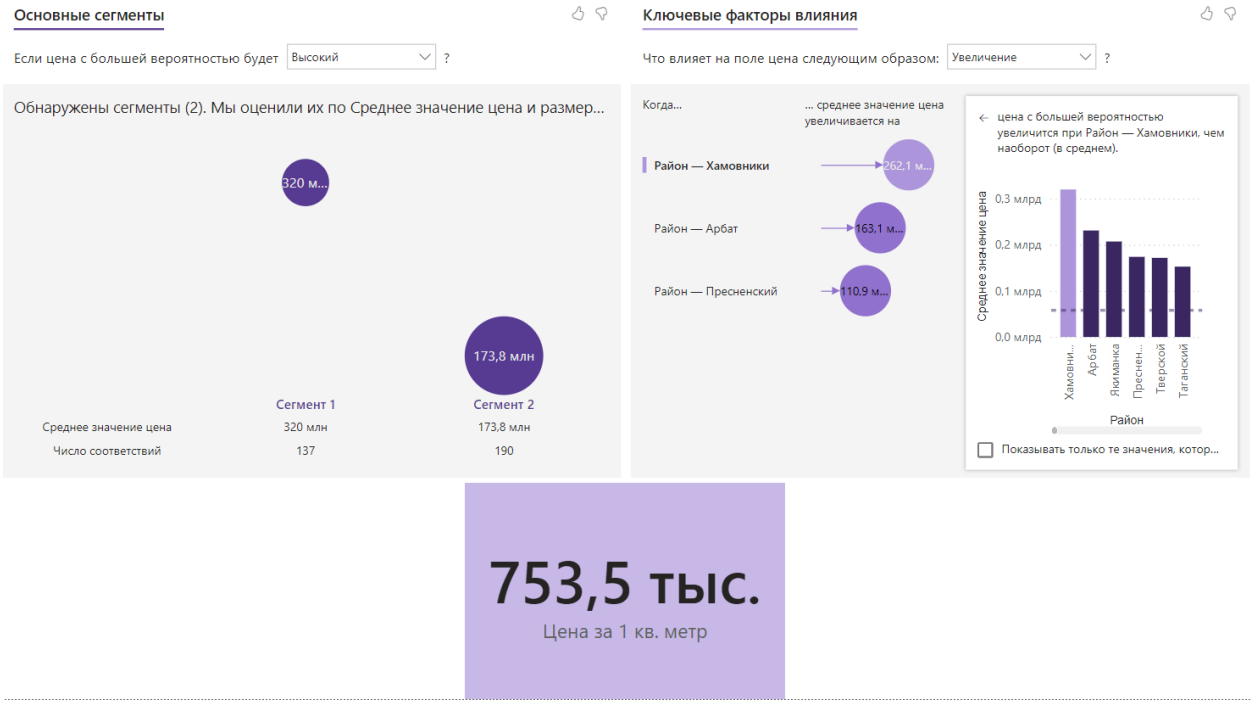
Второй график указывает на рост в цены в зависимости от количества комнат, он особо не изменяется, и всегда растет вверх при увеличении числа комнат.

Третий график указывает на число объявлений в зависимости от возраста здания, таким образом пик продаж пришелся на 2023 г.

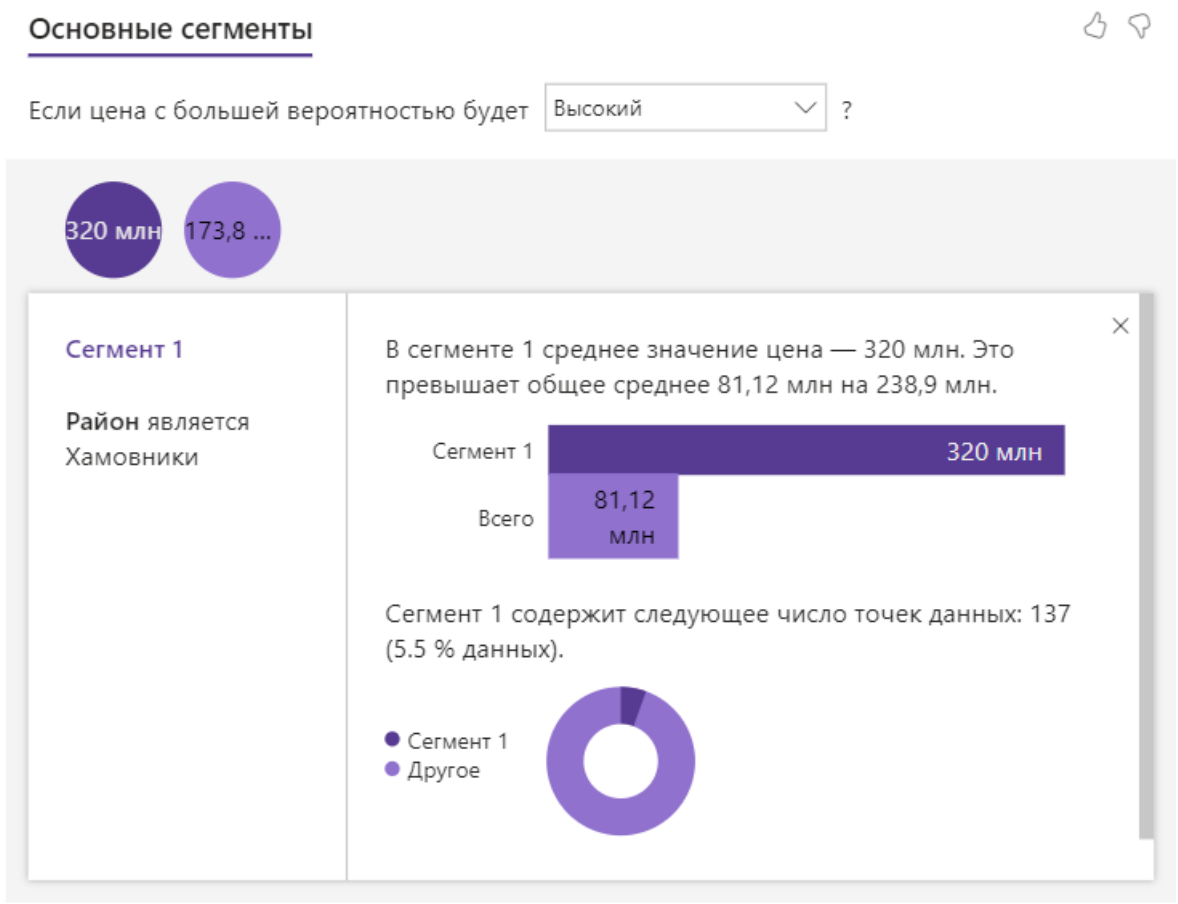
Четвертый график показывает, что если дом сдан допустим в 2024 г., то квартира будет стоить примерно 23 млн. рублей, пик цены пришел на 1901 г., обоснование тому, что жилье продается в историческом месте и имеет большое значение для культурного наследства, поэтому его цена крайне высока и она так сильно повлияла на этот показатель.

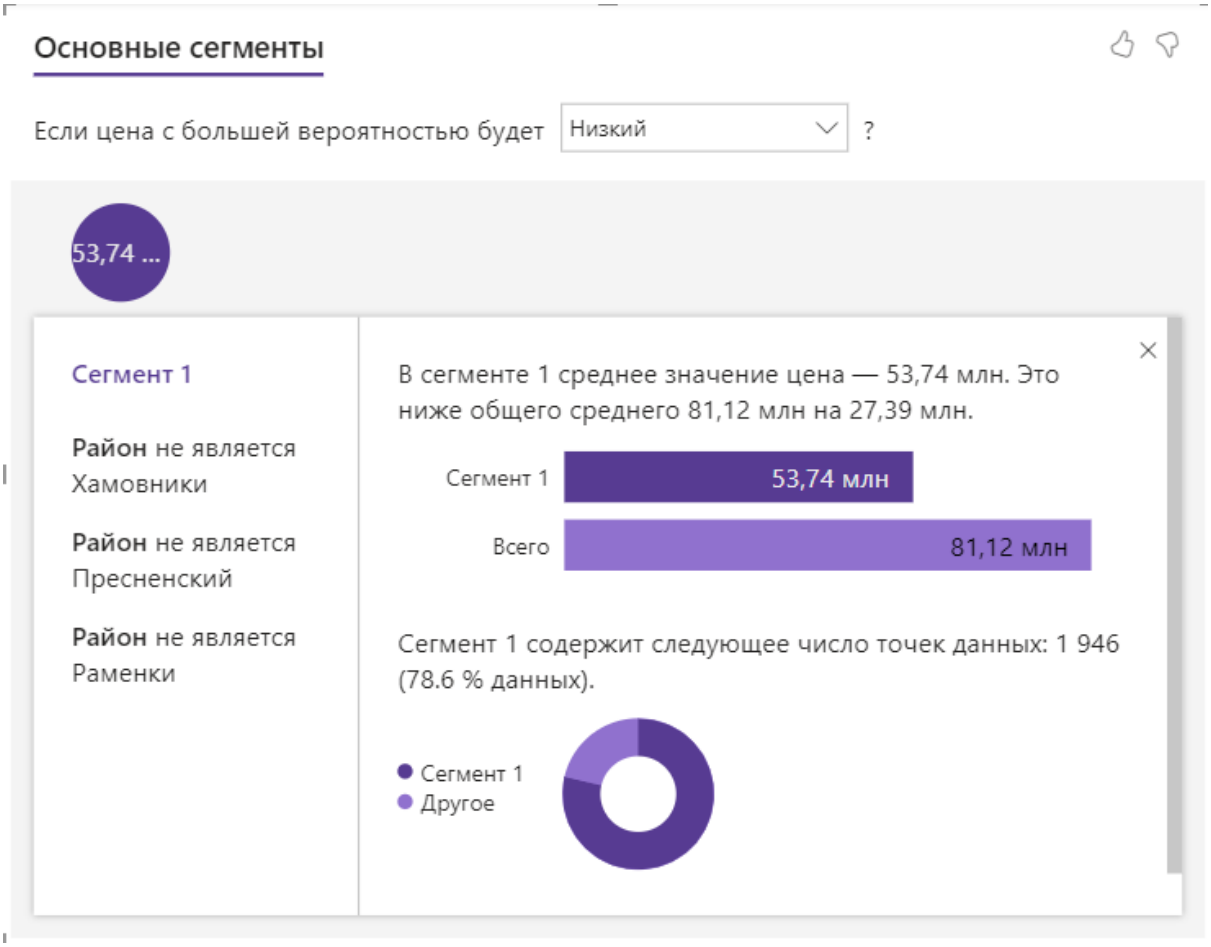
Теперь перейдем к следующем дашборду, данные в нем исключительно по Москве. Здесь высчитывается средняя стоимость кв. метра, в этом у меня возникли небольшие трудности так как подходящего визуального элемента и команды в программе нет, мне пришлось, используя язык программирования

DAX написать расчет нужного мне значения.



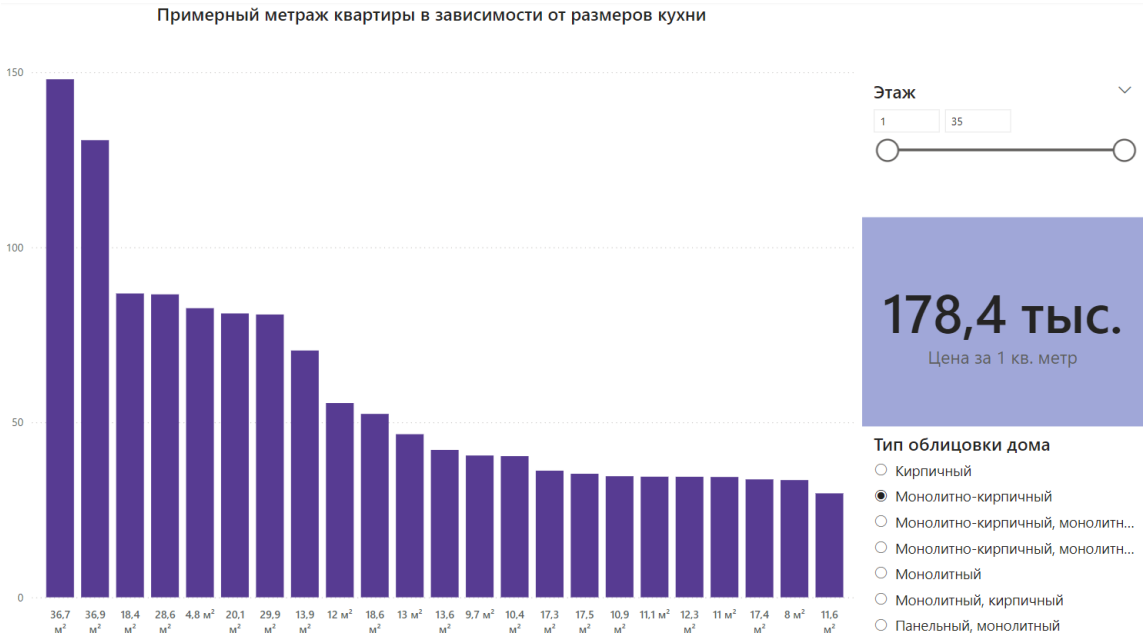
Также использованы встроенные модели для анализа данных в качестве визуального элемента, первая отвечает за счет чего будет высокой или наоборот низкой,





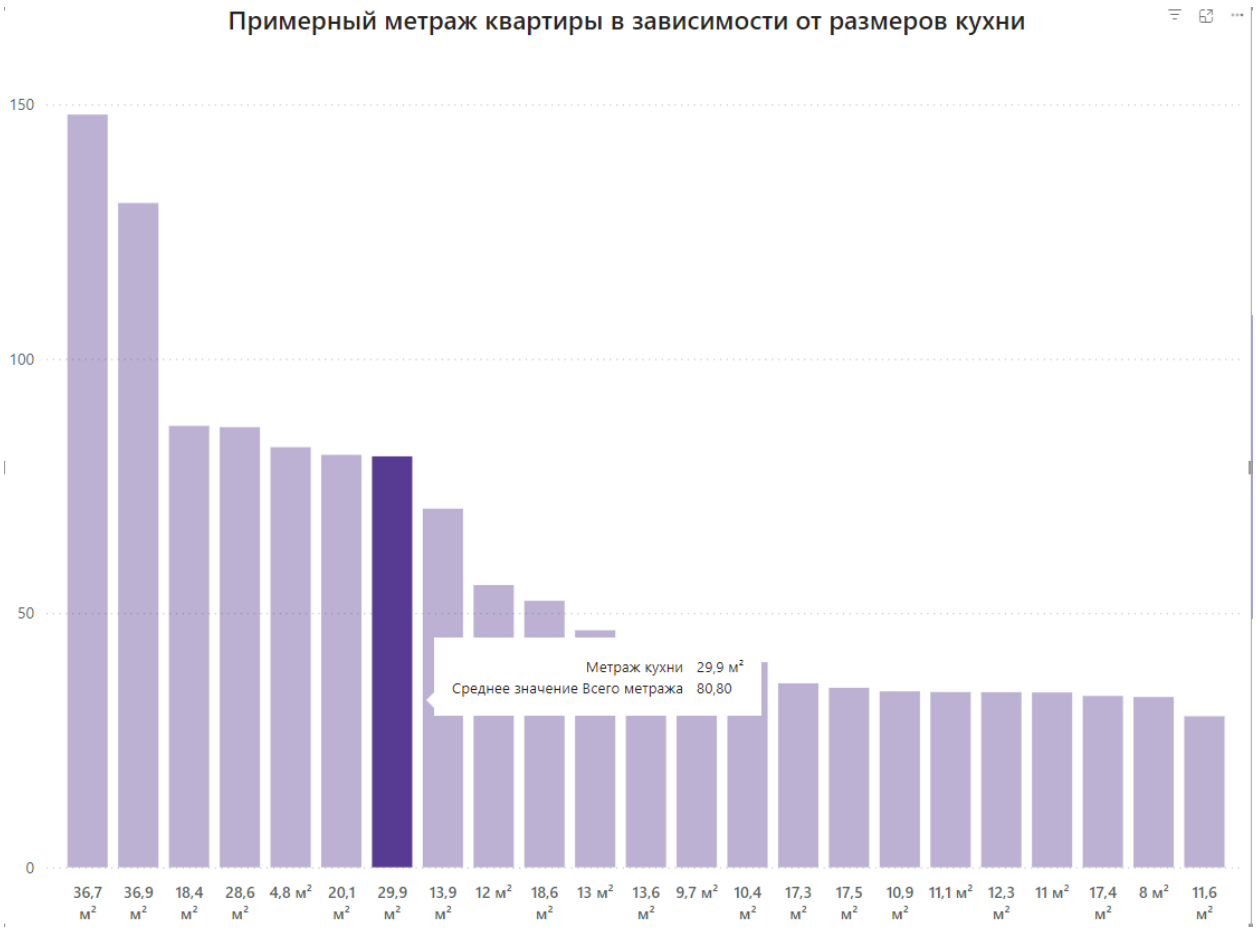
Вторая модель указывает на увеличение и уменьшение цены в зависимости в каком районе расположена квартира, а также строит график для сравнения с другими районами.

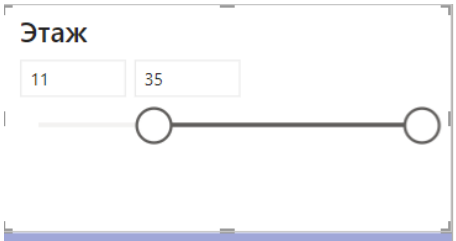
Четвертый дашборд работает с данными только по Московской области.

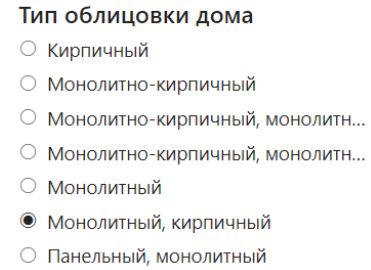


Здесь я выбрала довольно необычные способы зависимости цены от каких-либо факторов (площадь кухни, тип облицовки здания и этаж)

Для вывода средней цены кв. метра также пришлось писать свой расчет на языке DAX.







РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ

В ходе анализа были собраны данные, построена модель и создана визуализация, в которой были выявлены ключевые критерии для оценки стоимости квадратного метра недвижимости в Московском регионе. Основными факторами, влияющими на цену жилья, стали близость к станциям метро, расположение в крупных городах и наличие ремонта в квартире. Полученные результаты могут быть использованы для дальнейшего прогнозирования цен на недвижимость.