

Кейс компании «Самолет» (big data)

«Модель оценки цены квартиры на вторичном рынке по Московскому региону: Москва, Долгопрудный, Балашиха»

Актуальность задачи:

Оценка недвижимости - важная составляющая девелоперского бизнеса. Информация, о реальной цене квартиры исходя из рынка, интересна для покупателей продавцов, застройщиков, агентов и др.

Эту информацию можно использовать по-разному, в частности, у бизнеса есть потребность использовать такую модель для лидогенерации клиентов. Потенциальный клиент заходит на сайт, хочет оценить свою квартиру, вводит параметры: квартира, на 3 этаже, 3-х комнатная, дизайнерский ремонт. Затем нажимает кнопку «узнать цену», и, прежде чем дать ответ, мы хотим запросить у клиента номер телефона или адрес почты.

Для этого нужна модель оценки цены квартиры по Московскому региону.

Цель: Собрать данные и провести разведочный исследовательский анализ данных (EDA) для построения модели, которая будет оценивать цену квадратного метра недвижимости в Московском регионе (Москва и Московской области).

Процесс проделанной работы:

Пишем код для “парсинга” сайта Циан (Рисунок 1).

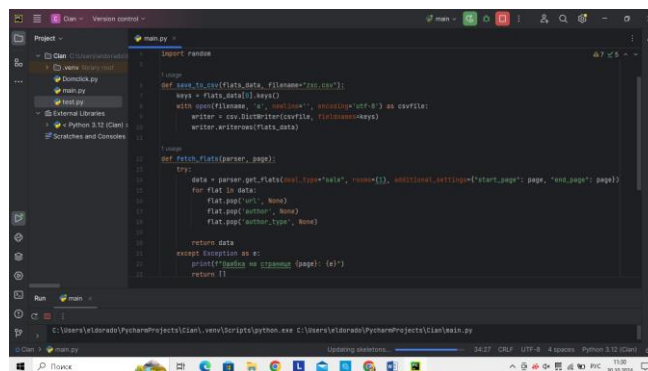
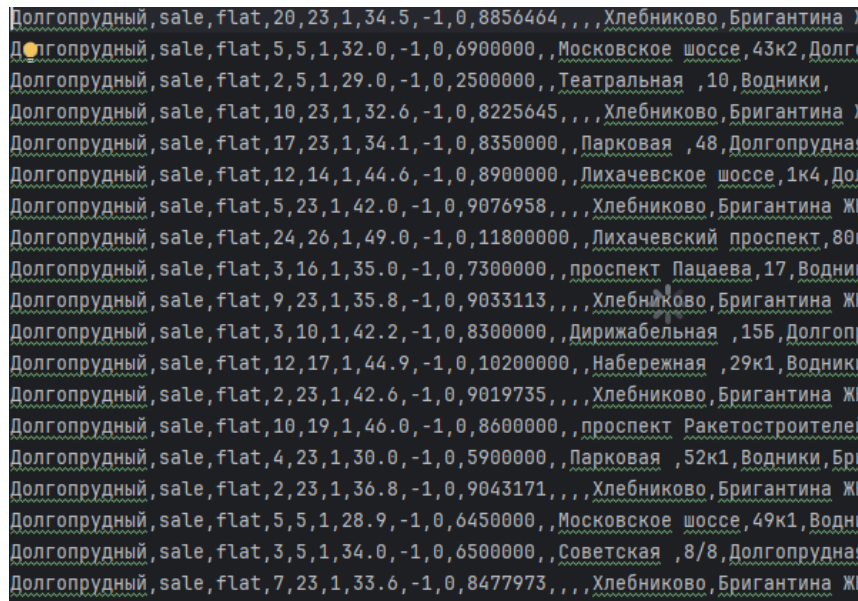


Рисунок 1 – Код

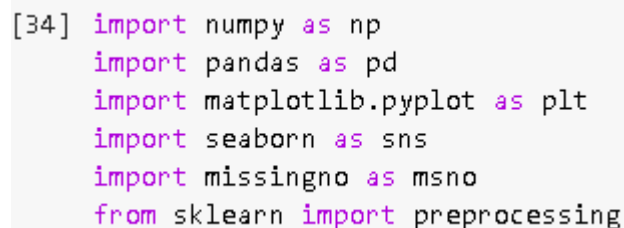
Загружаем данные в “.csv – файл” (Рисунок 2).



```
Долгопрудный,sale,flat,20,23,1,34.5,-1,0,8856464,,,Хлебниково,Бригантина Ж  
Долгопрудный,sale,flat,5,5,1,32.0,-1,0,6900000,,Московское шоссе,43к2,Долго  
Долгопрудный,sale,flat,2,5,1,29.0,-1,0,2500000,,Театральная,10,Водники,  
Долгопрудный,sale,flat,10,23,1,32.6,-1,0,8225645,,,Хлебниково,Бригантина Ж  
Долгопрудный,sale,flat,17,23,1,34.1,-1,0,8350000,,Парковая,48,Долгопрудная  
Долгопрудный,sale,flat,12,14,1,44.6,-1,0,8900000,,Лихачевское шоссе,1к4,Дол  
Долгопрудный,sale,flat,5,23,1,42.0,-1,0,9076958,,,Хлебниково,Бригантина Ж  
Долгопрудный,sale,flat,24,26,1,49.0,-1,0,11800000,,Лихачевский проспект,80к  
Долгопрудный,sale,flat,3,16,1,35.0,-1,0,7300000,,проспект Пацаева,17,Водник  
Долгопрудный,sale,flat,9,23,1,35.8,-1,0,9033113,,,Хлебниково,Бригантина Ж  
Долгопрудный,sale,flat,3,10,1,42.2,-1,0,8300000,,Дирижабельная,15Б,Долгоп  
Долгопрудный,sale,flat,12,17,1,44.9,-1,0,10200000,,Набережная,29к1,Водники  
Долгопрудный,sale,flat,2,23,1,42.6,-1,0,9019735,,,Хлебниково,Бригантина Ж  
Долгопрудный,sale,flat,10,19,1,46.0,-1,0,8600000,,проспект Ракетостроителей  
Долгопрудный,sale,flat,4,23,1,30.0,-1,0,5900000,,Парковая,52к1,Водники,Бри  
Долгопрудный,sale,flat,2,23,1,36.8,-1,0,9043171,,,Хлебниково,Бригантина Ж  
Долгопрудный,sale,flat,5,5,1,28.9,-1,0,6450000,,Московское шоссе,49к1,Водни  
Долгопрудный,sale,flat,3,5,1,34.0,-1,0,6500000,,Советская,8/8,Долгопрудная  
Долгопрудный,sale,flat,7,23,1,33.6,-1,0,8477973,,,Хлебниково,Бригантина Ж
```

Рисунок 2 – Выгруженные данные

Переходим в Google Colab, первым делом импортируем библиотеки (Рисунок 3)



```
[34] import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
import missingno as msno  
from sklearn import preprocessing
```

Рисунок 3 - Библиотеки

- numpy: Библиотека для работы с многомерными массивами и матрицами, а также для выполнения математических операций.
- pandas: Библиотека для анализа данных, предоставляющая структуры данных (например, DataFrame) и функции для работы с ними.
- matplotlib.pyplot: Библиотека для создания статических, анимационных и интерактивных графиков.
- seaborn: Библиотека для визуализации данных, основанная на matplotlib, с улучшенным интерфейсом и дополнительными функциями.
- missingno: Библиотека для визуализации пропусков в данных.

- `sklearn.preprocessing`: Модуль из библиотеки `scikit-learn`, содержащий функции для предварительной обработки данных, включая кодирование категориальных признаков.

Выводим таблицу (Рисунок 4).

```
df = pd.read_csv('home.csv')
df.columns = ['Город', 'Тип сделки', 'Тип недвижимости', 'Этаж квартиры', 'Сколько этажей в доме', 'Количество комнат', 'Общая площадь квартиры', 'Цена', 'Микрорайон', 'Улица', 'Дом', 'Станция метро или ид', 'Название жилого комплекса']
df.iloc[::]
```

	Город	Тип сделки	Тип недвижимости	Этаж квартиры	Сколько этажей в доме	Количество комнат	Общая площадь квартиры	Цена	Микрорайон	Улица	Дом	Станция метро или ид	Название жилого комплекса
0	Балашиха	sale	flat	7	16	1	42.30	7250000	NaN	Юбилейная	4к5	Железнодорожная	NaN
1	Балашиха	sale	flat	3	16	1	34.00	6000000	NaN	Жилгородок	9А	Ольино	NaN
2	Балашиха	sale	flat	15	22	1	36.39	6390771	NaN	NaN	к8	Некрасовка	Новоград Павлино
3	Балашиха	sale	flat	19	25	1	36.30	5300000	NaN	Яганова	9	Щёлковская	Пехра
4	Балашиха	sale	flat	2	16	1	32.00	4900000	NaN	Октябрьская	17	Железнодорожная	NaN
...
4517	Долгопрудный	sale	flat	3	9	2	49.50	8600000	мкр. Гранитный	Молодежная	20	NaN	NaN
4518	Долгопрудный	sale	flat	17	23	1	34.10	8350000	NaN	Парковая	48	Долгопрудная	Бригантина
4519	Долгопрудный	sale	flat	5	5	2	50.70	8850000	NaN	Восточная	39	Долгопрудная	NaN
4520	Долгопрудный	sale	flat	3	9	1	37.00	7500000	NaN	Академика Лаврентьева	5	Водники	NaN
4521	Долгопрудный	sale	flat	2	5	2	42.70	8600000	NaN	Комсомольская	11	Долгопрудная	NaN

4522 rows x 13 columns

Рисунок 4 - Таблица

Выводим количество строк и столбцов удаляем дубликаты (Рисунок 5).

```
In [6]:
df.shape

Out[6]:
(7978, 27)

In [36]:
#Удаляем дубликаты
df = df.drop_duplicates()

In [38]:
df.shape

Out[38]:
(7216, 20)
```

После удаления дубликатов осталось 7216 строк

Рисунок 5 – Удаляем дубликаты

Удаляем ненужные колонки (Рисунок 6).

```
df.drop(columns=['author', 'author_type', 'deal_type', 'url', 'accommodation_type', 'phone', 'house_number'], inplace=True)
```

Рисунок 6 - Вывод информации

Визуализация пропусков (Рисунок 7).

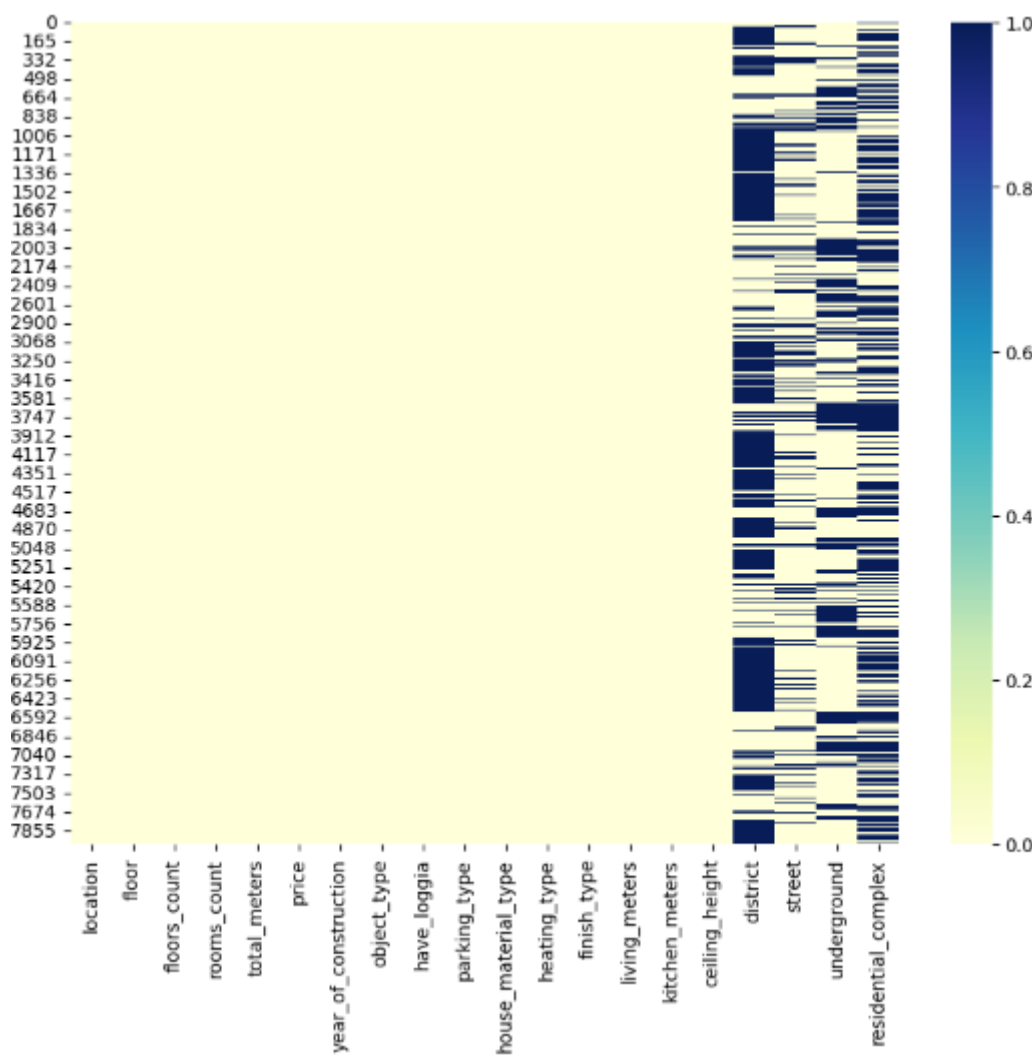


Рисунок 7 – Пропуски

Меняем значения '-1' на nan (Рисунок 8).

```
# "-1" не имеет смысла (нужно 0)
df.replace('-1', np.nan, inplace=True)

df.head(10)
```

ipython-input-44-bab6b10aec02>:2: SettingWithCopyWarning:
value is trying to be set on a copy of a slice from a DataFrame

see the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

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

	location	floor	floors_count	rooms_count	total_meters	price	year_of_construction	object_type	have_loggia	par
0	Москва	11	13	5	265.6	405328840	2024	Новостройка	NaN	
1	Москва	26	31	5	246.7	288063785	2008	Вторичка	3 балкона	Г
2	Москва	4	8	5	117.0	58500000	1939	Вторичка	1 балкон	
3	Москва	24	24	5	172.5	207000000	2024	Новостройка	NaN	
4	Москва	3	9	5	234.8	125000000	1999	Вторичка	1 лоджия, 2 балкона	Г
5	Москва	57	76	5	303.4	260000000	2008	Вторичка / Апартаменты	NaN	Г
6	Москва	5	5	5	85.0	19000000	NaN	Вторичка	2 балкона	
7	Москва	23	24	5	415.0	150000000	1998	Вторичка	2 лоджии	Г
8	Москва	4	16	5	217.9	205915500	2026	Новостройка	1 лоджия	Г
9	Москва	17	17	5	148.0	120000000	2022	Вторичка	NaN	

Рисунок 8 – Заменяем значения

Выводим ещё раз диаграмму с пустыми значениями (Рисунок 9).

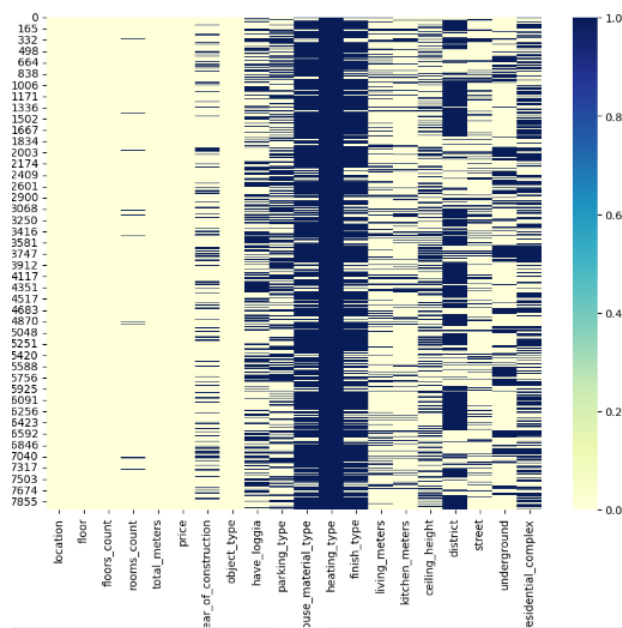


Рисунок 9 - Пропуски

Пропусков стало больше, потому что значения '-1' заменили на пропуски.

Удаляем столбцы, в которых много пропусков и строки с пропусками в параметре room_count (Рисунок 10).

```
#Удаление колонок и строчек, в которых пропуски
#Удаление строчек
df = df.dropna(subset=['rooms_count'])
#Удаление столбцов
df.drop(['house_material_type', 'heating_type', 'finish_type', 'district', 'residential_complex'], axis=1, inplace=True)
```

Рисунок 10 – Удаление пропусков

Если не указан балкон или ложа ставим 0 (Рисунок 11):

```
#Заменяем пропуски нулями
df['have_loggia'] = df['have_loggia'].fillna('0')
df['parking_type'] = df['parking_type'].fillna('0')
```

Рисунок 11 – Замена значений

Удаляем строки где цена не указана (Рисунок 12).

```
df.shape
```

```
(7122, 15)
```

```
df = df.dropna(subset=['price'])
```

```
df.shape
```

```
(7109, 15)
```

Рисунок 12 – Удаление пустых значений

Меняем тип данных с object на int или float (Рисунки 13-20)

- rooms_count

```
df['rooms_count'].unique()
#rooms_count has unique values не устраивает
```

```
array(['5', '1', '3', '2', '4', 'rooms_count'], dtype=object)
```

```
# НОЛЮБЛЕЖ ЭТУ СТРОЧКУ
df_rooms = df.loc[df['rooms_count'] == 'rooms_count']
df_rooms
```

	location	floor	floors_count	rooms_count	total_meters	price	y
2999	location	floor	floors_count	rooms_count	total_meters	price	

```
# УДОБЛЯЕМ ЭТУ СТРОЧКУ
df.drop(2999, inplace=True)
```

python-input-77-82c265dea6d7>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/10min.html#copy-on-write>
df.drop(2999, inplace=True)

```
# НЕПРАВИЛЬНЫЙ ПОДХОД
df['rooms_count'] = pd.to_numeric(df['rooms_count'], errors='raise')
df['rooms_count']
```

Рисунок 13 – rooms_count

```
df['floor'] = pd.to_numeric(df['floor'], errors='raise')
df['floor']
```

python-input-85-d322ca8cd328>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Please use `.loc[row_indexer, col_indexer] = value` instead
See the caveats in the documentation: <https://pandas.pydata.org/pandas-docs/stable/10min.html#copy-on-write>
df['floor'] = pd.to_numeric(df['floor'], errors='raise')

	floor
0	11
1	26
2	4
3	24
4	3
...	...
1973	2
1974	2
1975	7
1976	4
1977	7

108 rows × 1 columns

type: int64

```
df['floors_count'] = pd.to_numeric(df['floors_count'], errors='raise')
df['floors_count']
```

Рисунок 14 - floor и floors_count

- total_meters

```
df['total_meters'].unique()
```

```
array(['265.6', '246.7', '117.0', ..., '78.14', '79.38', '60.99'],  
      dtype=object)
```

```
df['total_meters'] = pd.to_numeric(df['total_meters'], errors='raise')  
df['total_meters']
```

Рисунок 15 - total_meters

- Высота потолков

```
df['ceiling_height'].unique()
```

```
array(['3\хаам', '3,1\хаам', '3,2\хаам', nan, '3,5\хаам', '3,65\хаам',  
      '4\хаам', '3,52\хаам', '3,6\хаам', '3,24\хаам', '2,82\хаам',  
      '3,08\хаам', '2,62\хаам', '2,7\хаам', '2,66\хаам', '2,63\хаам',  
      '2,72\хаам', '2,8\хаам', '2,74\хаам', '3,04\хаам', '2,75\хаам',  
      '2,5\хаам', '1,8\хаам', '2,9\хаам', '3,74\хаам', '2,85\хаам',  
      '2,6\хаам', '2,64\хаам', '2,65\хаам', '2\хаам', '2,52\хаам',  
      '2,68\хаам', '2,69\хаам', '3,75\хаам', '2,78\хаам', '3,05\хаам',  
      '2,54\хаам', '3,3\хаам', '3,25\хаам', '3,15\хаам', '3,12\хаам',  
      '2,95\хаам', '2,56\хаам', '2,76\хаам', '3,45\хаам', '2,77\хаам',  
      '2,88\хаам', '3,4\хаам', '2,55\хаам', '2,4\хаам', '2,84\хаам',  
      '2,53\хаам', '2,48\хаам', '2,87\хаам', '2,83\хаам', '4,4\хаам',  
      '2,58\хаам', '2,57\хаам', '2,73\хаам', '52\хаам', '2,26\хаам',  
      '2,3\хаам', '1,65\хаам', '3,02\хаам', '3,17\хаам', '4,5\хаам',  
      '2,98\хаам', '3,72\хаам', '2,97\хаам', '2,90\хаам', '3,35\хаам',  
      '3,27\хаам', '5,7\хаам', '4,2\хаам', '2,92\хаам', '3,01\хаам',  
      '2,67\хаам', '2,71\хаам', '0\хаам', '2,78000000000002\хаам',  
      '6\хаам', '3,43\хаам', '2,51\хаам', '5,2\хаам', '2,89\хаам',  
      '3,36\хаам', '3,7\хаам', '25\хаам', '9\хаам', '3,34\хаам',  
      '5,5\хаам', '5,1\хаам', '5\хаам', '4,55\хаам', '3,9\хаам',  
      '3,03\хаам', '3,14\хаам', '3,59\хаам', '2,61\хаам', '3,26\хаам',  
      '2,81\хаам', '2,2\хаам', '4,6\хаам', '3,06\хаам', '3,09\хаам',  
      '2,42\хаам', '3,23\хаам', '3,55\хаам', '3,31\хаам'], dtype=object)
```

```
df['ceiling_height'] = df['ceiling_height'].str.replace(r'\хаам', '', regex=True).str.replace(',', '.')  
df['ceiling_height'] = df['ceiling_height'].astype(float)  
df['ceiling_height']
```

Рисунок 16 - ceiling_height

- object_type

```
print(f'Уникальные значения: {df["object_type"].unique()}')
```

```
Уникальные значения: ['Новостройка' 'Вторичка' 'Вторичка / Апартаменты'  
Новостройка / Пентхаус' 'Новостройка / Апартаменты'  
Вторичка / Пентхаус']
```

```
# Оставляем значения Вторичка и новостройка и переводим их в 1 и 0
```

```
def values(value):  
    if value == 'Вторичка / Апартаменты':  
        return 'Вторичка'  
    elif value == 'Вторичка / Пентхаус':  
        return 'Вторичка'  
    elif value == 'Новостройка / Пентхаус':  
        return 'Новостройка'  
    elif value == 'Новостройка / Апартаменты':  
        return 'Новостройка'  
    else:  
        return value
```

```
df['object_type'] = df['object_type'].apply(values)
```

```
df['object_type'] = df['object_type'].map(dict(Вторичка = 1, Новостройка = 0))  
df['object_type']
```

Рисунок 17 - object_type меняем значения на 1 и 0


```

'1987', '2021', '2019', '2015', '1988', '2006', '2016', '2014',
'1993', '1990', '2020', '2011', '1983', '1984', '1961', '1968',
'2018', '1994', '1979', '2017', '2009', '1977', '1966', '1988',
'2012', '1974', '1965', '2007', '1985', 'Напишите автору', '1962',
'1972', '1970', '1996', '1959', '1986', '1982', '1929', '1975',
'1967', '1930', '1971', '2005', '2000', '1963', '1969', '1952',
'1997', '1978', '1964', '2003', '1954', '2001', '1958', '1960',
'1902', '1897', '1992', '1917', '2004', '1989', '1957', '1900',
'1973', '1981', '1991', '1995', 'Позвоните автору', '1955', '1949',
'1947', '1953', '1910', '1901', '1909', '1600', '1951', '1940',
'1938', '1956', '1948', '1936', '1931', '1950', '1934', '1928',
'1945', '1937', '1927', '1896', '1777', '2031', '1935', 'Аукцион',
'1941', '1932', '1926', '1943', '1904', '1915'], dtype=object)

#удалены не верные данные
values_to_filter = ['Напишите автору', 'Позвоните автору', 'Аукцион']
tupp = df['year_of_construction'].isin(values_to_filter)

df = df[~tupp]

df['year_of_construction'].unique()

array(['2024', '2008', '1939', '1999', nan, '1998', '2026', '2022',
       '2028', '2010', '2002', '2023', '2027', '1976', '2025', '2013',
       '1987', '2021', '2019', '2015', '1988', '2006', '2016', '2014',
       '1993', '1990', '2020', '2011', '1983', '1984', '1961', '1968',
       '2018', '1994', '1979', '2017', '2009', '1977', '1966', '1988',
       '2012', '1974', '1965', '2007', '1985', '1962', '1972', '1970',
       '1967', '1930', '1971', '2005', '2000', '1963', '1969', '1952', '1997', '1978',
       '1954', '2003', '1954', '2001', '1958', '1960', '1902', '1897',
       '1992', '1917', '2004', '1989', '1957', '1900', '1973', '1981',
       '1991', '1995', '1955', '1949', '1947', '1953', '1910', '1901',
       '1909', '1600', '1951', '1940', '1938', '1956', '1948', '1936',
       '1931', '1950', '1934', '1928', '1945', '1937', '1927', '1896',
       '1777', '2031', '1935', '1941', '1932', '1926', '1943', '1904',
       '1915'], dtype=object)

# Кол-во пропусков
df["year_of_construction"].isna().sum()

1220

df['year_of_construction'].describe()

```

Рисунок 18 - year_of_construction

```

df['living_meters'] = df['living_meters'].str.replace(r'\xa0+', '', regex=True).str.replace(',', '.')
df['living_meters'] = df['living_meters'].astype(float)
df['living_meters']

```

living_meters	
0	NaN
1	140.9
2	NaN
3	104.6
4	140.0
...	...
7973	28.4
7974	24.0
7975	24.0
7976	30.0
7977	NaN

7055 rows x 1 columns

dtype: float64

```

df['kitchen_meters'] = df['kitchen_meters'].str.replace(r'\xa0+', '', regex=True).str.replace(',', '.')
df['kitchen_meters'] = df['kitchen_meters'].astype(float)
df['kitchen_meters']

```

Рисунок 19 - living_meters и kitchen_meters

```
# КОДИРУЕМ
from sklearn import preprocessing

columns = ['have_loggia', 'street', 'underground', 'parking_type',
           'location']

for col in columns:
    le = preprocessing.LabelEncoder()
    df[col] = le.fit_transform(df[col])
    integer_mapping = {l: i for i, l in enumerate(le.classes_)}
```

Рисунок 20 - Кодлируем оставшиеся значения

Удаляем аномалии в ceiling_height и в year_of_construction (Рисунки 21-23).

```
# Удаляем 3-х самых больших значений
df.loc[df['ceiling_height'].nlargest(3).index]
```

	location	floor	floors_count	rooms_count	total_meters	price	year_o
1100	Балашиха	1	2	2	58.3	5700000	
3776	Клин	5	5	1	31.9	3850000	
3948	Щёлково	1	9	2	49.0	5449000	

```
# Удаляем 2-х самых маленьких значений
df.loc[df['ceiling_height'].nsmallest(2).index]
```

	location	floor	floors_count	rooms_count	total_meters	price	year
2606	Котельники	2	5	2	47.59	12611350	
1500	Балашиха	13	15	3	73.60	11300000	

```
df.drop(2606, inplace=True)
df.drop(1500, inplace=True)
df.drop(3776, inplace=True)
```

Рисунок 21 - ceiling_height

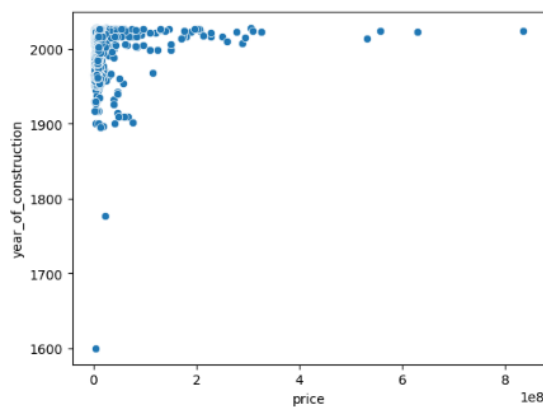


Рисунок 22 - year_of_construction от цены

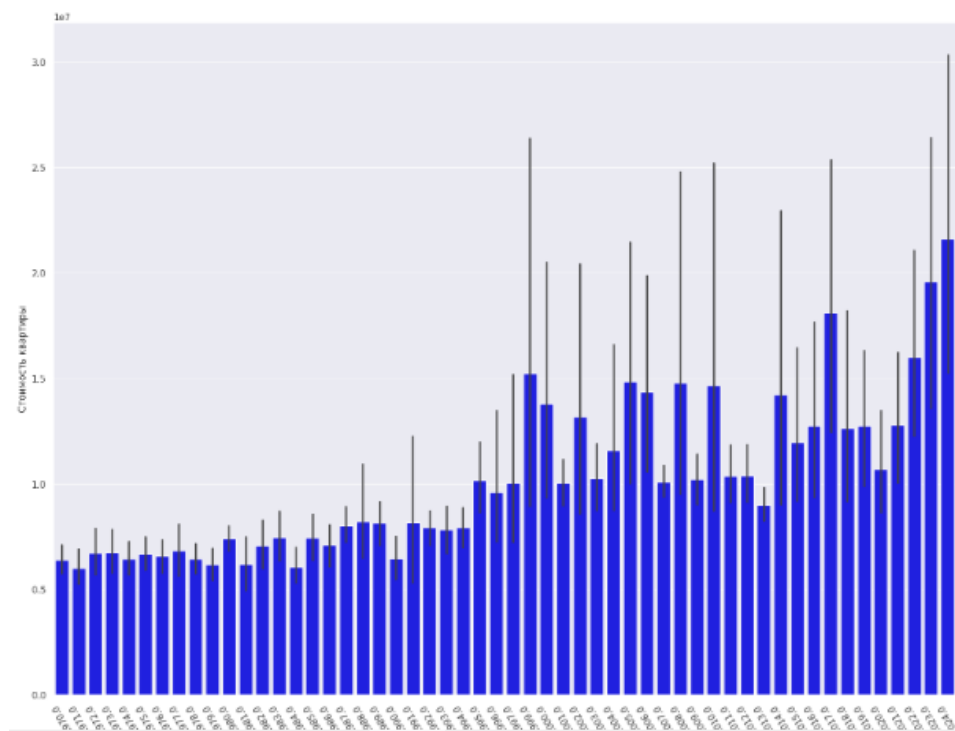


Рисунок 23 -График year_of_construction от price

Матрица корреляции (Рисунок 24).

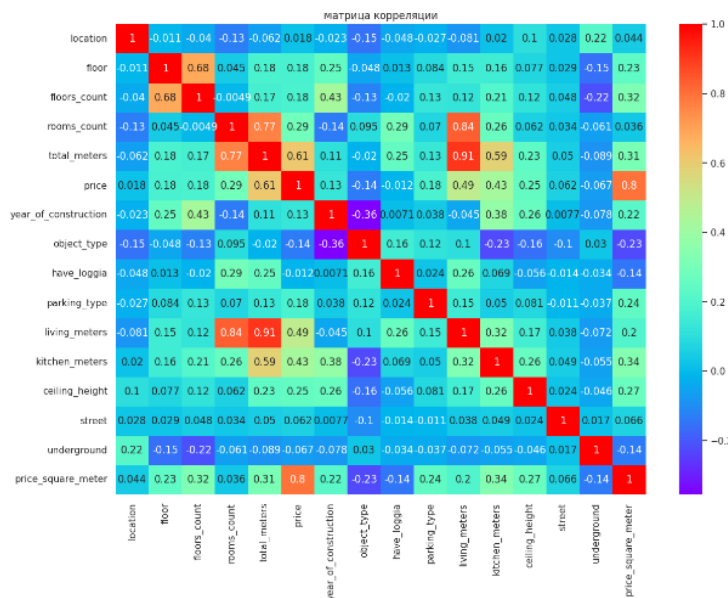


Рисунок 24 - Матрица корреляции

Графики для наглядности (Рисунок 25).

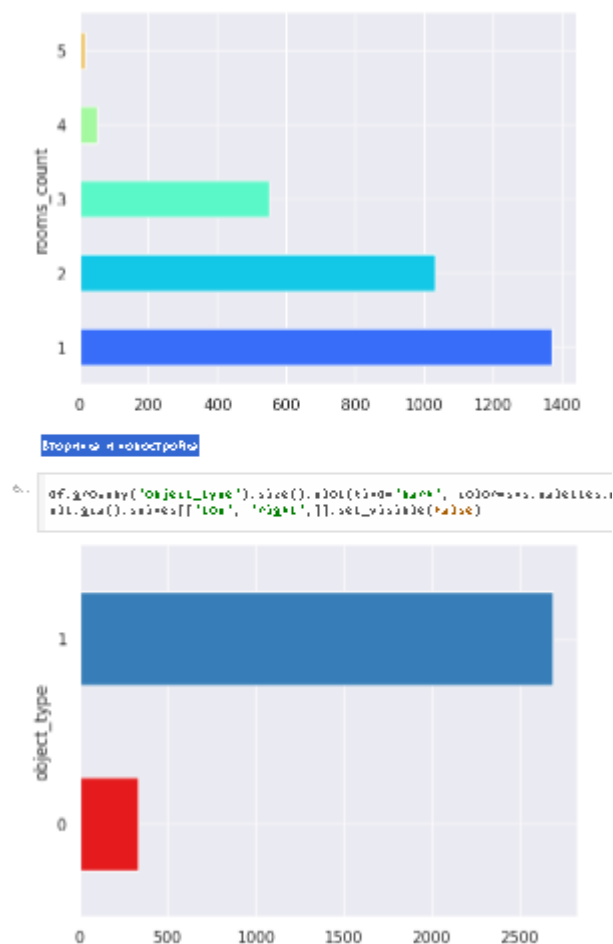


Рисунок 25 - Графики

Вывод по данным о зависимости цены квартиры от различных факторов можно сформулировать следующим образом:

Цена квартиры является многогранным показателем, который зависит от сочетания нескольких ключевых факторов. Основные из них включают площадь, этаж, состояние квартиры, расположение, наличие инфраструктуры и возраст здания.

1. Площадь: Прямо пропорциональна цене — чем больше площадь, тем выше стоимость.

2. Этаж: Влияние этажа может быть, как положительным, так и отрицательным в зависимости от особенностей здания и предпочтений покупателей.

3. Состояние квартиры: Квартиры с современным ремонтом значительно увеличивают свою стоимость по сравнению с объектами, требующими ремонта.

4. Расположение: Местоположение играет решающую роль, особенно в крупных городах, где центральные районы традиционно имеют более высокие цены.

5. Инфраструктура: Близость к социальным объектам, таким как школы и магазины, также способствует повышению цен на квартиры.

6. Возраст здания: Новые дома часто имеют более высокую стоимость благодаря современным технологиям и материалам.

Таким образом, для оценки стоимости квартиры необходимо учитывать комплекс этих факторов. Использование статистических методов, таких как множественная линейная регрессия, позволяет более точно предсказать цену на основе имеющихся данных, что полезно как для покупателей, так и для продавцов на рынке недвижимости.