

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

Отчёт по кейсу «Самолёт»:

Работу выполнил: Студент
группы № ИСП-21
Ежов Денис

Долгопрудный, 2024

Введение

В данном отчёте рассматриваются выводы, полученные с аналитической работы над данными в области «Квартиры в Московской области, Новой Москве и Москве».

Цель

Собрать данные и произвести аналитическую работу над ними для будущих работ, например, создание модели на основе выводов.

Задачи

- Используя открытые источники собрать список данных.
- На основе полученной информации произвести удаление ненужных данных, дополнение необходимых, выявление аномалий и их блокировка.
- Визуализация данных при помощи, как минимум, двух инструментов для подобных задач. Нахождение взаимосвязей между данными или их полное отсутствие, усреднённых показателей для уверенного отчёта.

Основная часть

Для выполнения основной задачи, существует небольшой выбор источников, откуда собирать данные, мною был выбран интернет-ресурс «Циан». При помощи скриптов, написанных на языке Python и библиотеке CianParser было получено свыше десяти тысяч объявлений в нужных регионах.

К следующей задаче подходит такое начало, как соединение собранных данных в одну таблицу при помощи написанной функции с библиотекой Pandas.

После сбора всей информации воедино и уборки дубликатов, нужно узнать, какого типа наши данные (рис.1), так как отталкиваясь от типа данных, мы будем применять разные методы к их сортировке.

Теперь мы убираем -1 как базовое не собранное значение и смотрим, какие данные у нас смогли собраться (рис.2) при помощи библиотеки missingno. Как мы можем увидеть, object_type и heating_type практически нигде не указываются, значит мы вынуждены их удалить, так как строить анализ будет невозможно.

Далее отсеиваем ненужные данные и форматируем некоторые столбцы, чтобы их было легче анализировать с помощью сторонних инструментов (рис.3). После выполнения объёмной чистки данных, нужно проверить их состояние – смотрим внутрь файла и бегло проверяем на аномалии, в случае их отсутствия приступаем к кодовой проверке данных (рис.4-7).

После полной очистки данных вручную и программно можем приступать к сбору графиков/аналитической работе при помощи библиотеки `matplotlib` для вывода графических изображений. Например будет 4 графика:

1. Цена за m^2 по городам.
2. Цена за m^2 в зависимости от материала, используемого при постройке здания.
3. Количество объявлений по городам.
4. Количество объявлений по годам постройки здания.

Для первых и последних двух графиков будем использовать один метод подсчёта данных.

Первый метод – отбор цены за квадратный метр по трафаретному коду и запись в csv-файл(рис.8).

Второй метод – использование встроенной функции `.value_counts()` и запись в csv-файл для дальнейшей обработки(рис.9).

В итоге получаем графики(рис.10), на основе которых уже можно проводить анализ, но мы перейдём к составлению графиков на Power BI.

Power BI – максимально удобный инструмент для составления графиков и аналитики данных.

Для первого графика мы выбираем данные `price`, меняем сумму на «среднее» и включаем их в график, на другую ось ставим «`year_of_construction`». Выбираем тип графика и получается примерный график со средней ценой квартиры, в зависимости от года его постройки(рис.11).

Второй график будет содержать в себе среднюю цену квартиры, в зависимости от материала здания(рис.12), просто вместо года постройки ставим тип материала. Добавлю к этим данным среднюю цену по виду отделки(рис.13). UPD: все графики были заменены на один dashboard(рис.14).

Аналитика данных

Благодаря выведенным графикам, можно сделать выводы, что цена в основном зависит от типа отделки, материала дома, города. От года постройки зданий зависит лишь их количество на рынке и количество комнат во время СССР, а на цену никак не влияет.

Заключение

В результате аналитической работы были собраны, отсортированы, почищены данные, построены удобные для анализа графики, благодаря которым получилось выявить не маловажные критерии в оценивании стоимости недвижимости в Московской Области, Москве и Новой Москве. Основными факторами, оказывающими влияние на стоимость, выявились тип отделки, материала здания и расположение. Полученные данные могут быть использованы для дальнейшей разработки прогностических моделей.

Открываем наши драгоценные данные

```
df = pd.read_csv('11K.csv')
```

Смотрим форму и колонки наших данных

```
print(df.dtypes)  
df.shape
```

author	object
author_type	object
url	object
location	object
deal_type	object
accommodation_type	object
floor	float64
floors_count	float64
rooms_count	float64
total_meters	object
price	float64
year_of_construction	object
object_type	float64
house_material_type	object
heating_type	float64
finish_type	object
living_meters	object
kitchen_meters	object
phone	float64
district	object
street	object
house_number	object
underground	object
residential_complex	object
dtype:	object
(10582, 24)	

(рис.1)

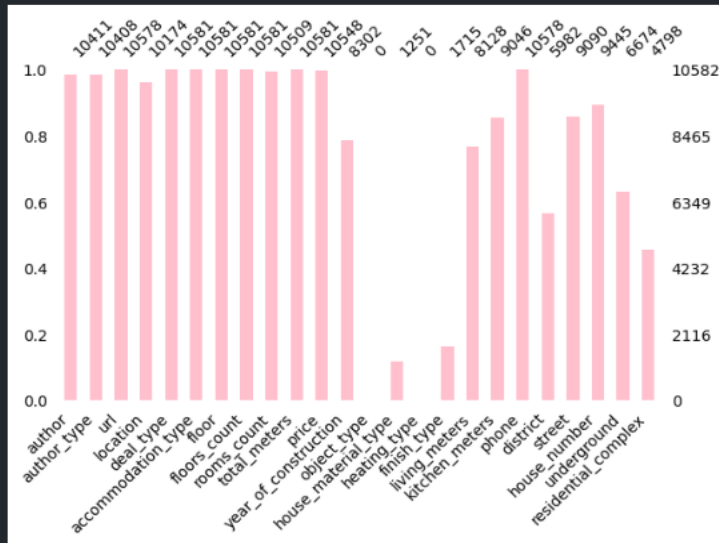
Чистим от отрицательных значений

```
df = df.replace(-1,np.nan) # Убираем все виды плохих или не собранных данных
df = df.replace("-1",np.nan) # Убираем все виды плохих или не собранных данных
df = df.replace(-1.0,np.nan) # Убираем все виды плохих или не собранных данных
df = df.replace("-1.0",np.nan) # Убираем все виды плохих или не собранных данных
df = df.replace("❖❖❖",0) # Убираем все виды плохих или не собранных данных
df.to_csv("half_11K.csv", index=False) # Тут запись в отдельный файл, дабы поэтапно отслеживать, на каком моменте появляются аномальные действия
# Ещё мы заходим в сохранённый файл и меняем ❖❖❖ на 0, дабы избежать ошибок (сделал отдельно)
```

Чеким корректность данных

```
msn.bar(df, figsize=(7,4), fontsize=10, color=(1, 0.75, 0.8))
```

<Axes: >



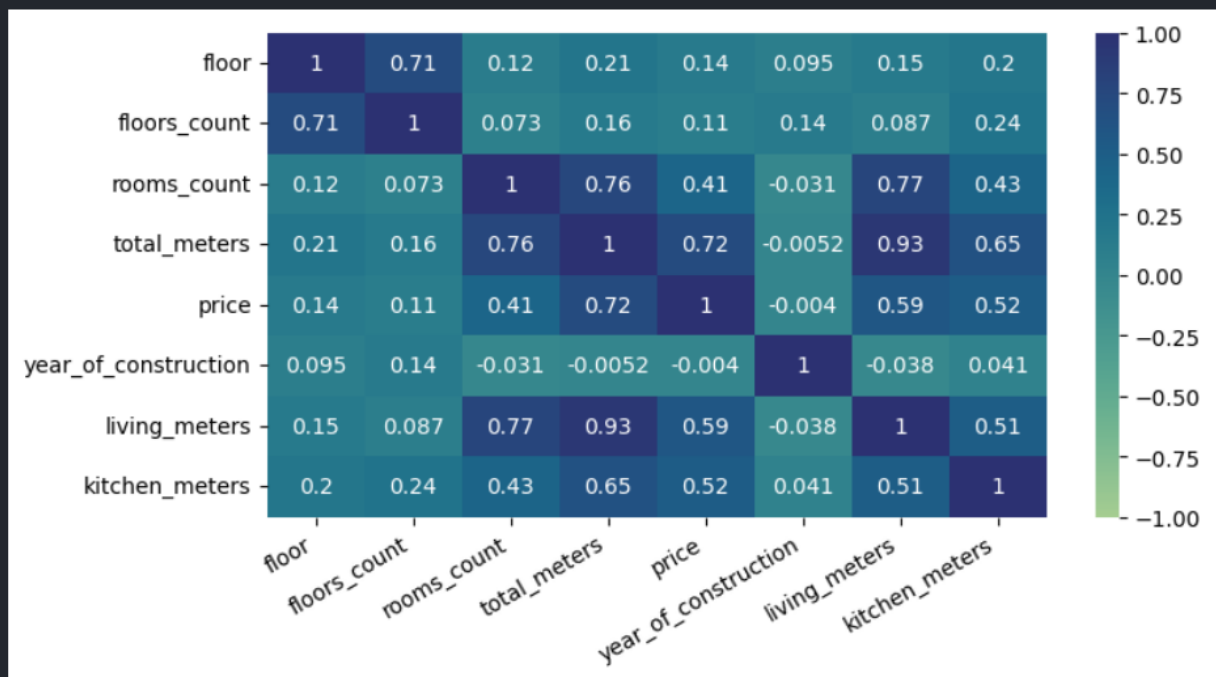
(рис.2)

```
df = pd.read_csv('half_11K.csv')
list = ["Напишите автору", "Залоговая недвижимость", "Аукцион", "Позвоните автору", "Подписаться на дом"]
for obj in list: df = df.replace(obj, np.nan) # Стираем
list1 = ["total_meters", "living_meters", "kitchen_meters"] # Данные, которые мы очистим от метров в ква
del df['author'] # Не надо
del df['author_type'] # Не надо
del df['accommodation_type'] # Это тип хаты
del df['deal_type'] # Тип сделки, у нас только продажа
del df['residential_complex'] # Много пропущено и не особо надо
del df['heating_type'] # Подогрев, нигде не указан
del df['object_type'] # Чет вообще не нужная штука
del df['phone'] # звонить для аналитики не нужно
list = ["price", "year_of_construction", "floor", "floors_count", "rooms_count"] # Создаём лист с данными,
for obj in list:
    df[obj] = df[obj].fillna(0) # Заполняем пустоты нулями
    df[obj] = df[obj].astype(int) # Оформляем INT для чистоты разума
for obj in list1: df[obj] = pd.to_numeric(
    df[obj].str.replace(',', '.').apply(lambda x: x[:-3] if pd.notna(x) else np.nan),
    errors='coerce'
).astype('float64')
df.to_csv("tret'_11K.csv", index=False) # Различия между cleaned_11K и tret'_11K - это ручной отброс п
```

(рис.3)

```
# Связи числовых данных
plt.figure(figsize=(8,4))
sns.heatmap(df.corr(numeric_only=True), cmap='crest', annot=True, vmin=-1, vmax=1)
plt.xticks(rotation=30, ha="right")
plt.show()
```

✓ 0.2s



(рис.4)

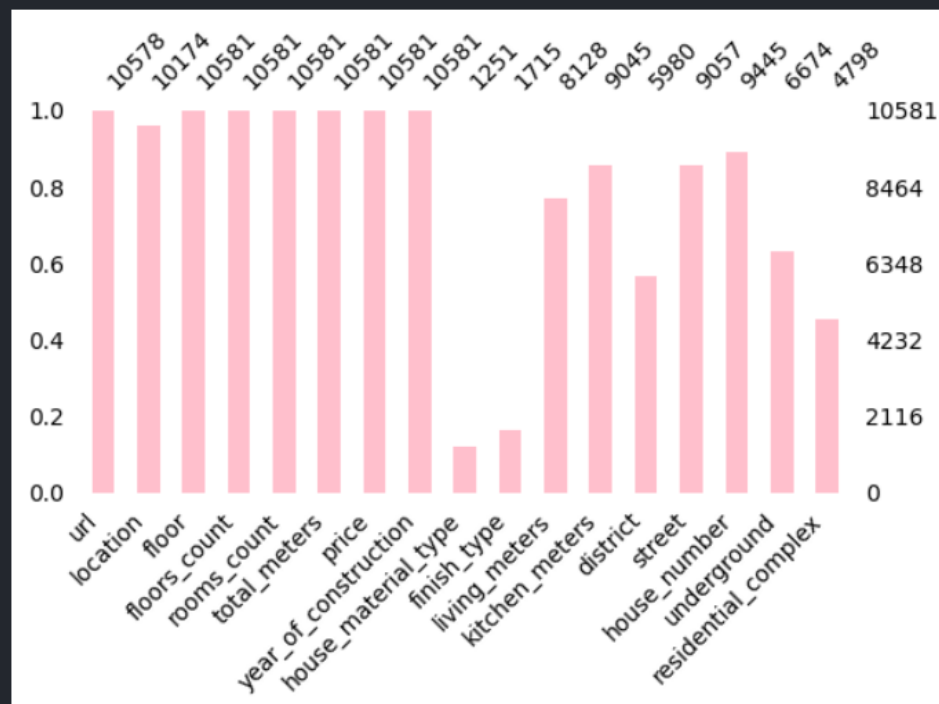
```
# Проверка на отсутствующие данные в процентах
for col in df.columns:
    pct_missing = np.mean(df[col].isnull())
    print('{} - {}'.format(col, round(pct_missing*100)))
```

```
url - 0%
location - 4%
floor - 0%
floors_count - 0%
rooms_count - 0%
total_meters - 0%
price - 0%
year_of_construction - 0%
house_material_type - 88%
finish_type - 84%
living_meters - 23%
kitchen_meters - 15%
district - 43%
street - 14%
house_number - 11%
underground - 37%
```

(рис.5)

```
# Проверка на отсутствующие данные в цифрах
msn.bar(df, figsize=(6,3), fontsize=10, color=(1, 0.75, 0.8))
plt.show()
```

✓ 0.2s



(рис.6)

```
print(df.dtypes)
df.shape
```

```
url                object
location           object
floor              int32
floors_count       int32
rooms_count        int32
total_meters       float64
price              int32
year_of_construction  int32
house_material_type object
finish_type        object
living_meters      float64
kitchen_meters     float64
district           object
street             object
house_number       object
underground        object
dtype: object

(10582, 16)
```

(рис.7)

```

df = pd.read_csv('cleaned_11K.csv')

list_of_cities = df['location'].unique()

def price_for_meter(location):
    city = df[df['location']==location]
    city_price = city['price'].sum();    cleaned_data = city['total_meters'].sum()
    return round(city_price/cleaned_data,2)

with open("dash_info_fifth.csv", 'w', newline='', encoding='UTF-8') as csvfile: # Создаем таблицу, чтобы у
    fieldnames = ['city', 'price_for_meter'];    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()
    for city in list_of_cities: writer.writerow({'city': city, 'price_for_meter': price_for_meter(city)})
# Потом вручную удаляем строчку "nan,nan"
df = pd.read_csv('cleaned_11K.csv')

list_of_cities = df['location'].unique()

def price_for_meter(location):
    city = df[df['location']==location]
    city_price = city['price'].sum();    cleaned_data = city['total_meters'].sum()
    return round(city_price/cleaned_data,2)

with open("dash_info_fifth.csv", 'w', newline='', encoding='UTF-8') as csvfile: # Создаем таблицу, чтобы удобнее было вынимать данные
    fieldnames = ['city', 'price_for_meter'];    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()
    for city in list_of_cities: writer.writerow({'city': city, 'price_for_meter': price_for_meter(city)})
# Потом вручную удаляем строчку "nan,nan"

```

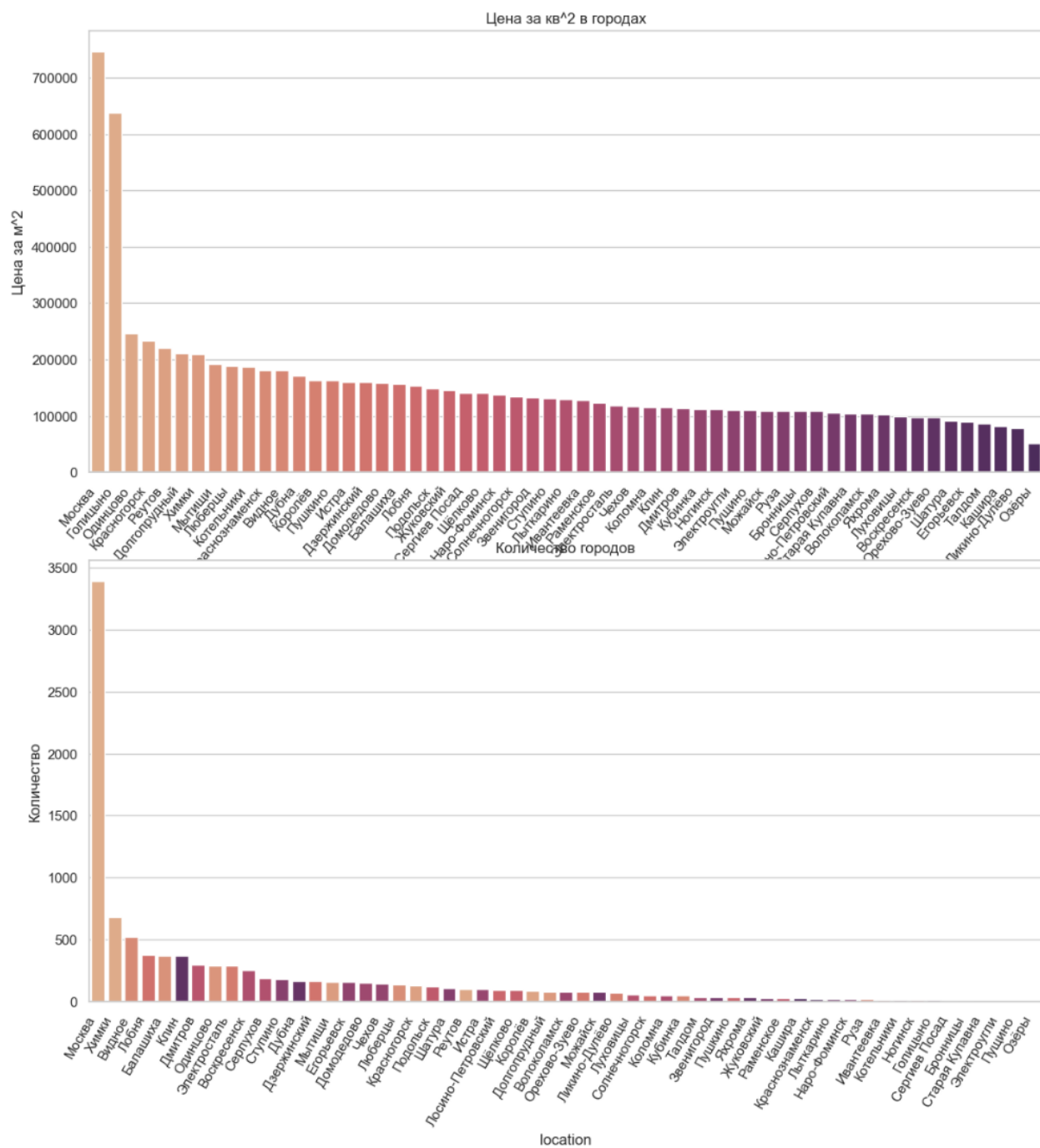
(рис.8)

```

df['year_of_construction'] = df['year_of_construction'].fillna(-1)
df['year_of_construction'] = df['year_of_construction'].astype(int)
df['year_of_construction'].value_counts().to_csv('years_count.csv')

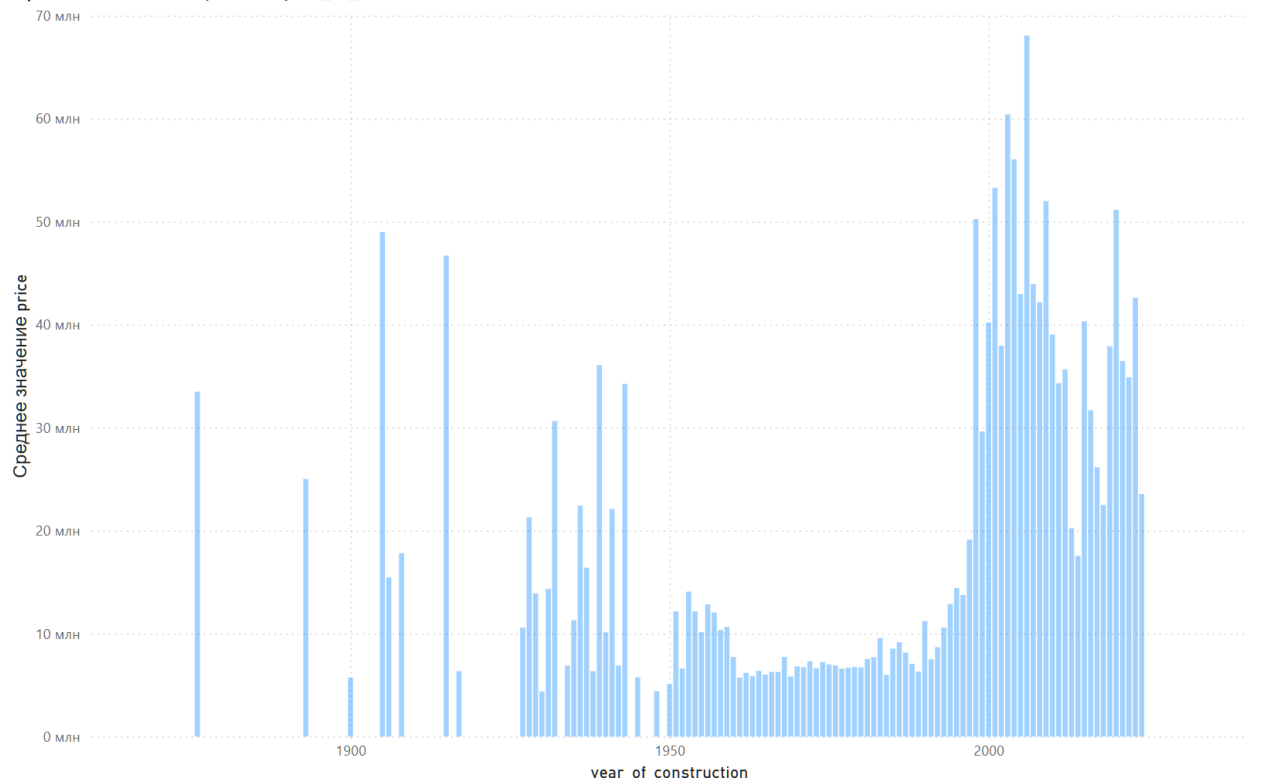
```

(рис.9)



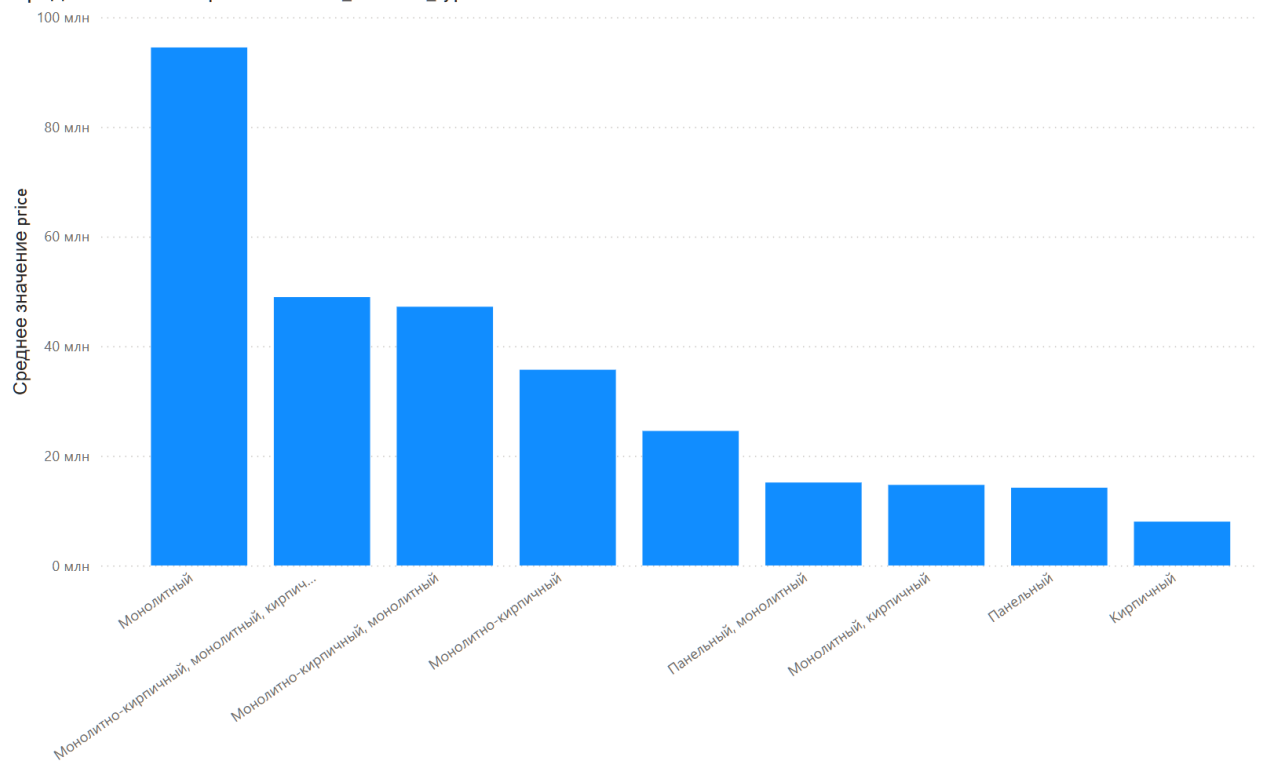
(рис.10)

Среднее значение price по year_of_construction



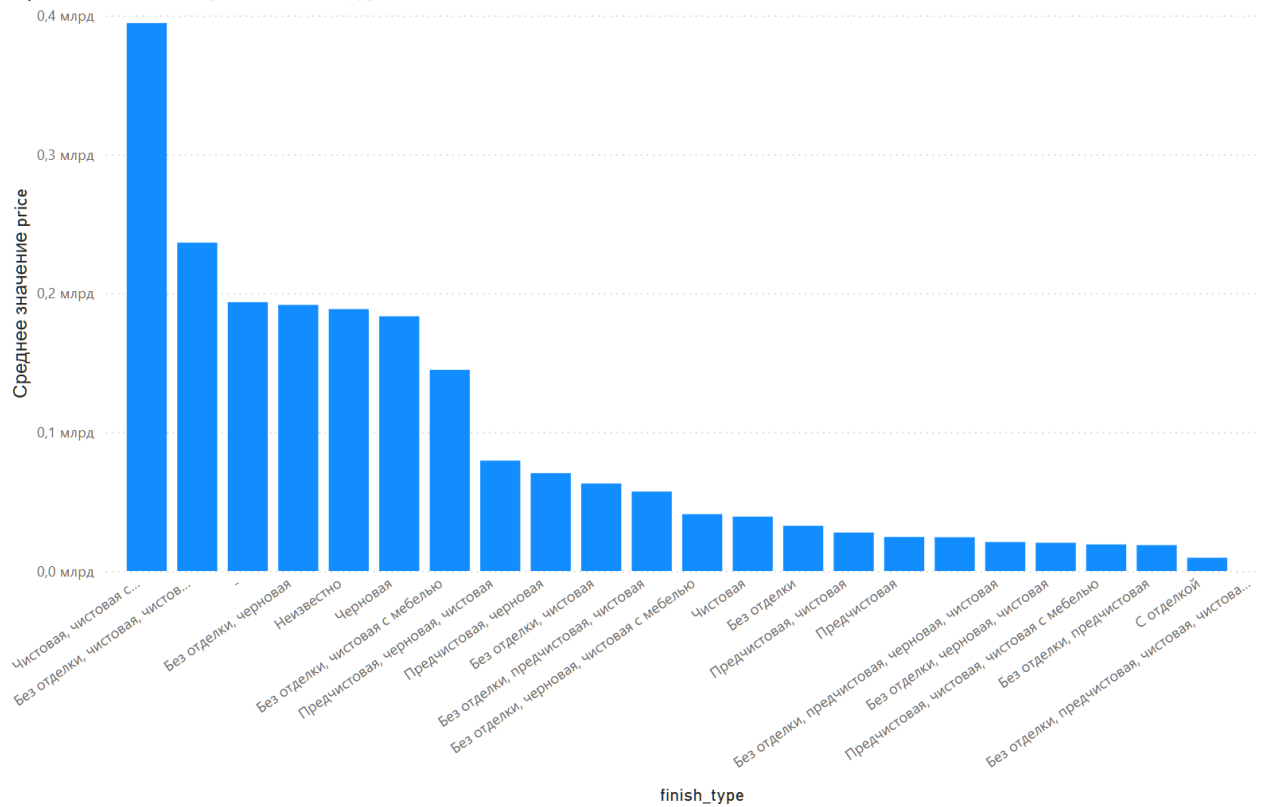
(рис.11)

Среднее значение price по house_material_type

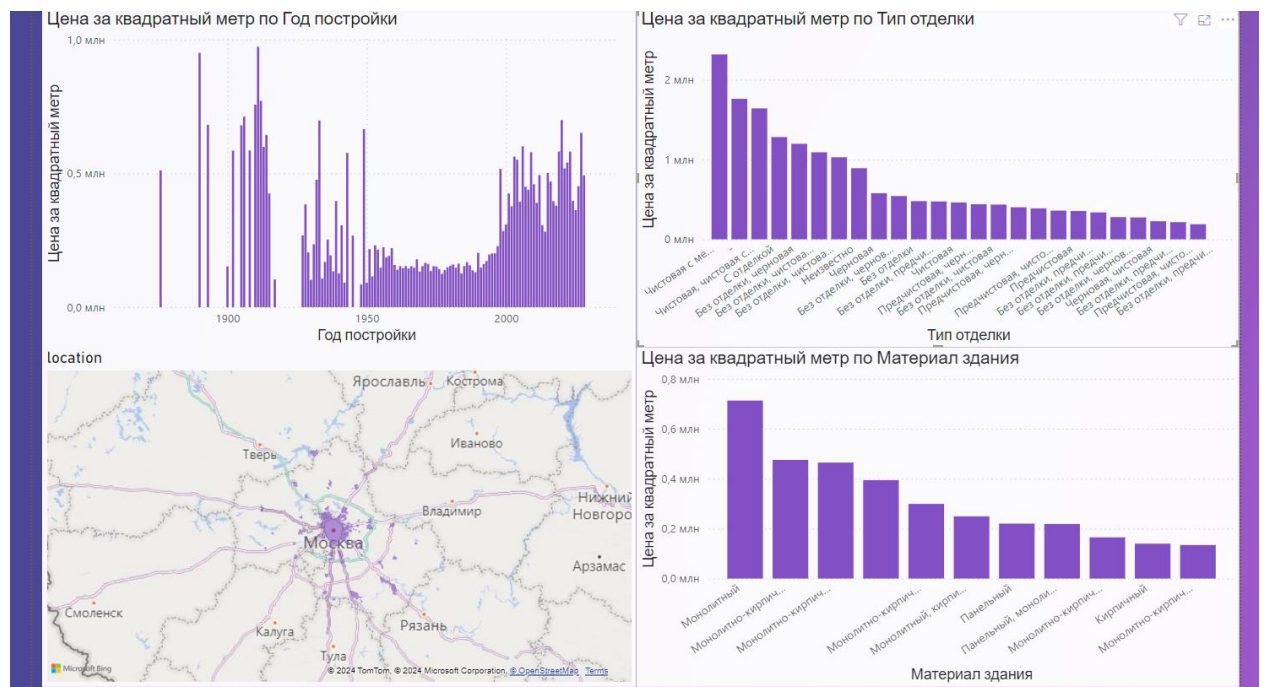


(рис.12)

Среднее значение price по finish_type



(рис.13)



(рис.14)