Государственное бюджетное профессиональное образовательное учреждение Московской области

«Физико-технический колледж»

# Аналитический отчёт

Работу выполнила: студент группы № ИСП-22 Серобян Кима

Долгопрудный, 2024

**Введение**

Оценка недвижимости является ключевым аспектом девелоперского бизнеса. Информация о реальной стоимости квартир представляет интерес для различных участников рынка, включая покупателей, продавцов, застройщиков и агентов.

**Цель** исследования заключается в сборе данных и проведении разведочного анализа данных для разработки модели, способной оценивать стоимость квадратного метра недвижимости в Московском регионе (включая Москву и Московскую область).

**Актуальность** данной темы обусловлена тем, что оценка стоимости квадратного метра недвижимости играет важную роль в анализе рынка, принятии обоснованных решений, развитии бизнеса и проведении исследований.

Задачи исследования включают:

* Составление списка факторов, существенно влияющих на цену квадратного метра жилой площади, на основе открытых источников и личного опыта;
* Проведение парсинга данных о квартирах на продажу с платформ, таких как Циан, и других;
* Подготовка данных для анализа: проверка на наличие пропусков, выбросов и ошибок, а также обработка выявленных аномалий (удаление или заполнение);
* Проведение Исследовательского Анализа Данных, включая построение распределения основных параметров, визуализацию взаимосвязей между ними и определение признаков, оказывающих наибольшее влияние на целевую переменную.

**Методология**

Применяемые инструменты и технологии:

* Библиотеки для парсинга: Cianparser
* Среда разработки для парсинга и формирования (сбора датасета): Visual Studio Code
* Библиотеки : Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn

**Сбор Информации**

Мы устанавливаем библиотеку Cianparser и в первую очередь редактируем файл page.py. Это необходимо для того, чтобы метод get\_flats() мог более эффективно извлекать дополнительную информацию из объявлений о продаже.

Изменённый файл page.py:

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,

            # "ceiling\_height": -1,

            # "ceiling\_height": -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))

        # ch = self.offer\_page\_soup.select\_one('[data-name="OfferSummaryInfoItem"] p:nth-of-type(10)').get\_text()

        # page\_data["ceiling\_height"] = ch

        # it = self.offer\_page\_soup.select\_one('[data-name="OfferSummaryInfoItem"] p:nth-of-type(10)').get\_text()

        # page\_data["ceiling\_height"] = it

        # et = self.offer\_page\_soup.select\_one('[data-name="OfferSummaryInfoItem"] p:nth-of-type(14)').get\_text()

        # page\_data["have\_loggia"] = et

        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

        # et = self.offer\_page\_soup.select\_one('[data-name="OfferSummaryInfoItem"] p:nth-of-type(2)').get\_text()

        # page\_data["heating\_type"] = et

        spans = self.offer\_page\_soup.select("span")

        for index, span in enumerate(spans):

            # if "Тип жилья" == span.text:

            #     page\_data["object\_type"] = spans[index + 1].text

            if "Тип дома" == span.text:

                page\_data["house\_material\_type"] = spans[index + 1].text

                time.sleep(5 + random.uniform(0, 5))

            # if "Отопление" == span.text:

            #     page\_data["heating\_type"] = spans[index + 1].text

            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\_\_()

Парсинг выполняется с помощью следующего кода:

parser = cianparser.CianParser(location="Дмитров")

data = parser.get\_flats(deal\_type="sale", rooms=(2), additional\_settings={"start\_page": 1, "end\_page": 54}, with\_extra\_data=True, with\_saving\_csv=True)

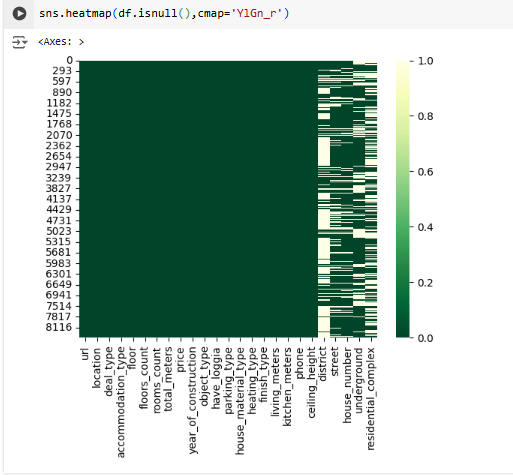
Значения для параметров location (город) и rooms (количество комнат) изменяются в зависимости от того, какие объявления необходимо спарсить. При каждом запуске кода создается файл в формате CSV, в который записываются собранные данные. Для объединения всех этих файлов в один DataFrame мы устанавливаем и импортируем библиотеку pandas, считываем все CSV-файлы с помощью функции read\_csv() и объединяем их с помощью метода concat().

Загружаем данные из файла, проверяем его размеры, удаляем дублирующиеся записи и снова отображаем размеры таблицы.

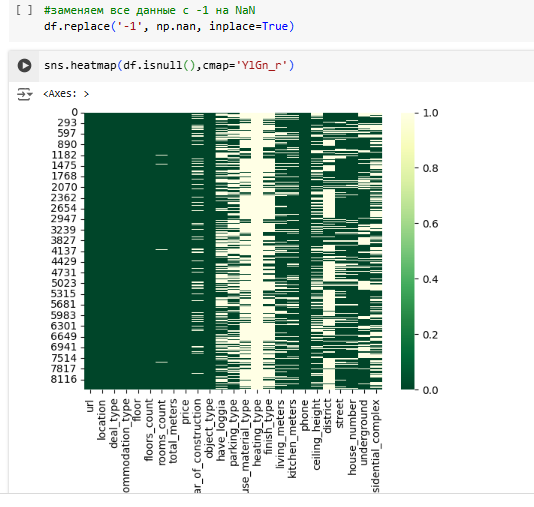


На данном этапе в таблице содержится 27 колонок. Первые две из них содержат информацию о авторе объявления (его имя и статус), которая не нужна для анализа, поэтому мы их удалим. После этого выведем информацию о таблице.

На тепловой карте нам показывает пропуски



Необходимо заменить все значения, равные «-1», на NaN и затем отобразить тепловую карту, которая покажет количество пропусков в колонках.



После того как мы заменили все данные с -1 на NaN. Видим что пропусков много. Следовательно начинаем с ними работать.

Удаляем столбцы «heating\_type, 'house\_material\_type', 'finish\_type», поскольку в них содержится большое количество пропусков.

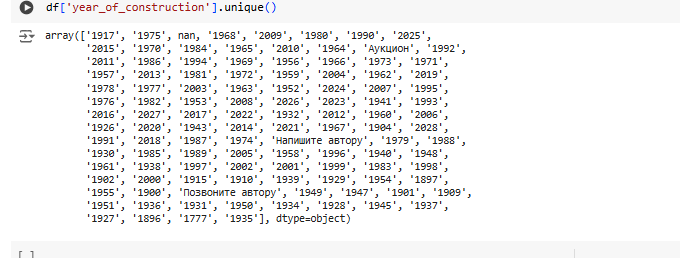
Также убираем колонки "housenumber" и "residentialcomplex". В адресе (местоположении) оставляем только метро и улицу. Колонку с номером телефона удаляем, так как она не требуется для анализа.

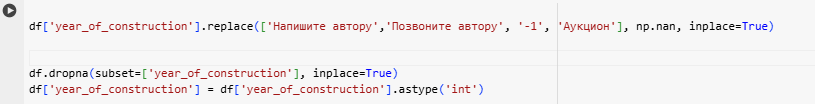
 Все данные в районе пустые, поэтому заполняем их данные из города.

Заполняем пропуски в каждом столбце медианными

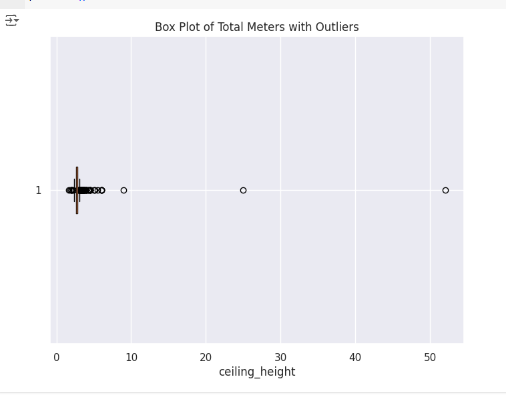


Затем мы займемся столбцом «year\_of\_construction»

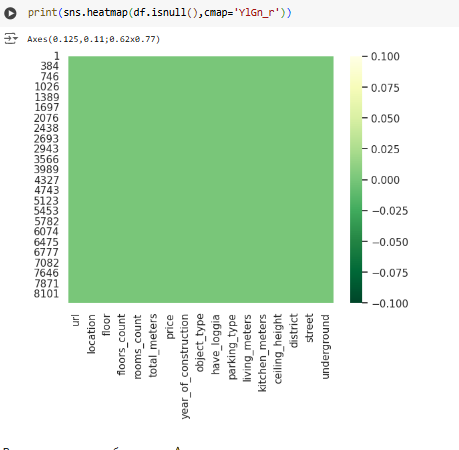


Удаляем строки, содержащие NaN

Видим ыбросы по колонке по потолку

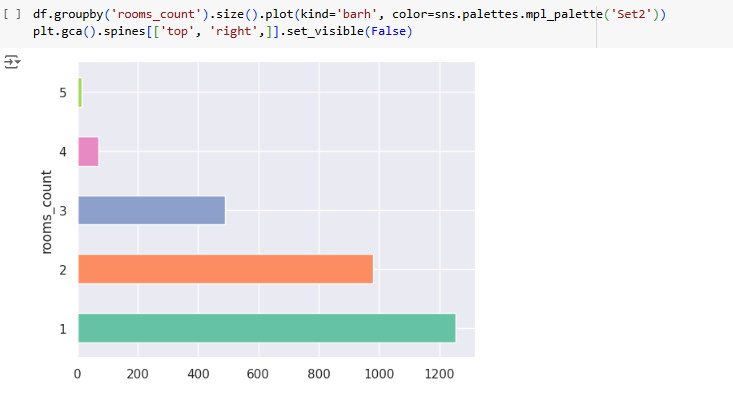
Следовательно их тоже удаляем



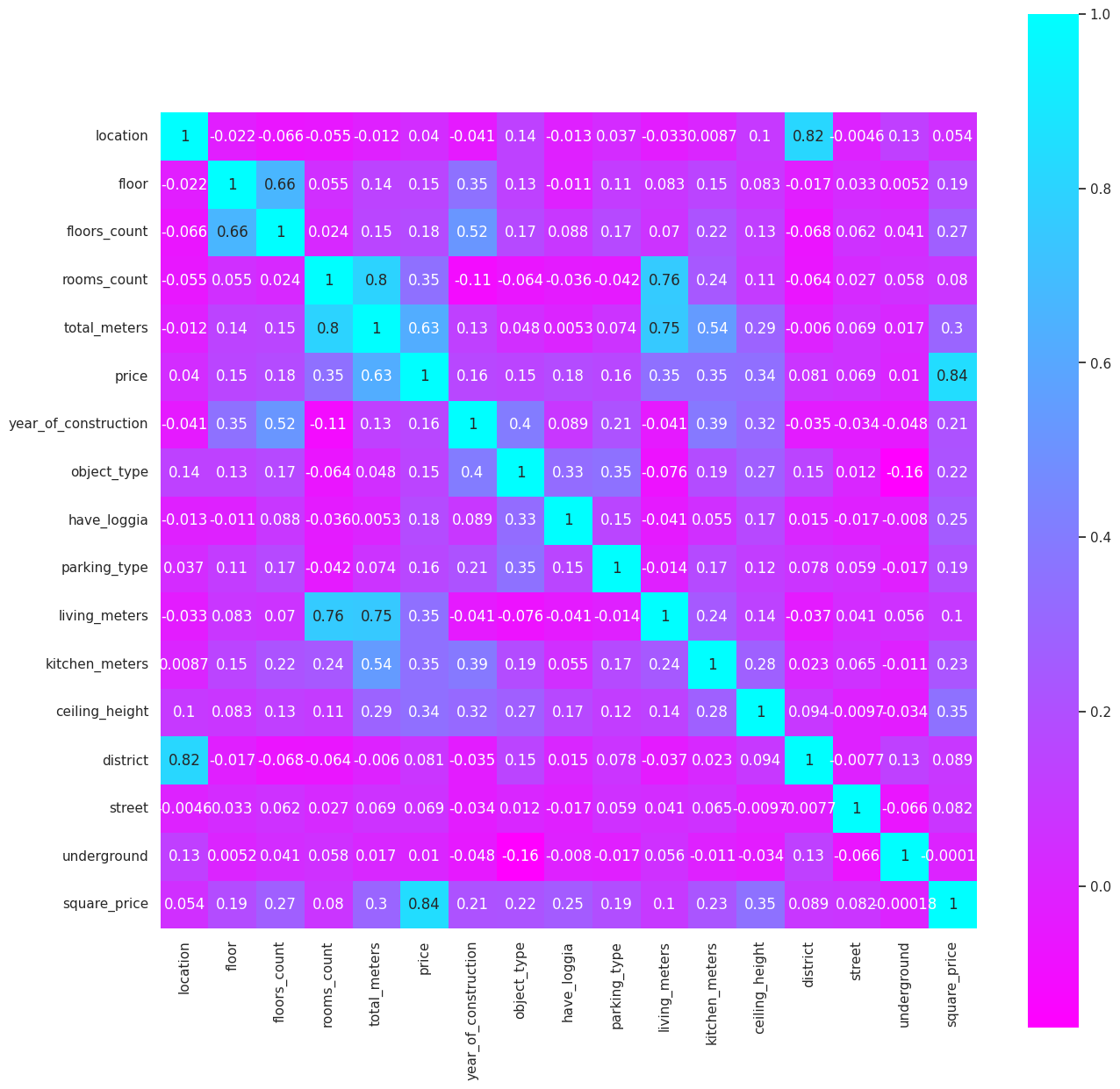
 Видим что пропусков больше нет!!!!

# Графики и анализ

Смотрим количество квартир по комнатам



На графике видно что, наибольшее количество однокомнатных квартир, а наименьшее — пятикомнатных

Матрица корреляции

**Заключение**

Цена за квадратный метр квартиры зависит от множества факторов, включая этаж и этажность здания, общую стоимость квартиры, различные площади (общую, кухонную и жилую), год постройки, высоту потолков, а также такие характеристики, как расположение в районе и наличие парковки.