Geekbrains

**Аналитика данных жилой недвижимости в городе Екатеринбург, с использованием датасета, полученного парсингом веб-сайта недвижимости, написанном на языке Python**

Программа: Аналитик

Специализация: Data Engineer

Чадов Денис Игоревич

г. Кыштым

2024 г.

[Введение 4](#_Toc177343609)

[1. Актуальность темы 4](#_Toc177343610)

[2. Цели и задачи работы 5](#_Toc177343611)

[3. Объект и предмет исследования 6](#_Toc177343612)

[4. Структура работы 7](#_Toc177343613)

[Глава 1 Теоретическая часть 8](#_Toc177343614)

[1. Обзор технологий веб-скрейпинга 8](#_Toc177343615)

[1.1. Понятие веб-скрейпинга и его применение 8](#_Toc177343616)

[1.2. Этические аспекты парсинга 9](#_Toc177343617)

[1.3. Инструменты веб-скрейпинга: сравнение популярных библиотек 11](#_Toc177343618)

[2. Изучение библиотеки BeautifulSoup 14](#_Toc177343619)

[2.1. Основные возможности 14](#_Toc177343620)

[2.2. Установка 15](#_Toc177343621)

[3. Анализ данных о недвижимости 16](#_Toc177343622)

[3.1. Изучение сайтов недвижимости 16](#_Toc177343623)

[3.2. Типовые данные, собираемые с сайтов недвижимости для последующей оценки и анализа 17](#_Toc177343624)

[3.3. Методы и инструменты для анализа данных 18](#_Toc177343625)

[4. Визуализация данных с помощью библиотек Matplotlib и Seaborn 19](#_Toc177343626)

[4.1. Основы визуализации данных 19](#_Toc177343627)

[4.2. Сравнение библиотек Matplotlib и Seaborn 20](#_Toc177343628)

[4.3. Типы графиков, применяемых для анализа данных о недвижимости 22](#_Toc177343629)

[Глава 2 Практическая часть 24](#_Toc177343630)

[1. Выбор сайта и сред разработки 24](#_Toc177343631)

[2. Разработка приложения для парсинга данных с сайта недвижимости 26](#_Toc177343632)

[3. Сбор и обработка данных 28](#_Toc177343633)

[4. Анализ собранных данных 30](#_Toc177343634)

[5. Визуализация результатов анализа 37](#_Toc177343635)

[Заключение 45](#_Toc177343636)

[1. Подведение итогов работы 45](#_Toc177343637)

[2. Основные выводы и рекомендации 46](#_Toc177343638)

[3. Перспективы и совершенствования исследования 49](#_Toc177343639)

[Список используемой литературы 51](#_Toc177343640)

# Введение

1. Актуальность темы

В современном мире цифровизация затрагивает все сферы нашей жизни, включая рынок недвижимости. С каждым годом интерес к онлайн-платформам для покупки, продажи и аренды жилья растет, что делает доступ к данным о недвижимости более актуальным, чем когда-либо. Развиваются и совершенствуются сайты недвижимости, улучшается их наполнение информацией для потенциальных клиентов. В то же время, данных становится слишком много, чтобы можно было проанализировать их обычными способами, по-старинке. Поэтому, парсинг сайтов недвижимости становится очень полезным инструментов, т.к. позволяет обрабатывать и анализировать большие объемы информации с веб-сайтов, что предоставляет множество возможностей для глубинного анализа тенденций на рынке недвижимости.

Перечислим основные достоинства данного инструмента:

Во-первых, парсинг информации из открытых источников позволяет собрать актуальные и разнообразные данные о ценах, характеристиках объектов, их расположении и других факторах, влияющих на стоимость недвижимости. Это особенно важно в условиях быстро меняющегося рынка, когда традиционные методы получения данных могут оказаться неэффективными и устаревшими.

Во-вторых, анализ полученных данных делает возможным выявление скрытых закономерностей и трендов, что может помочь как покупателям, так и продавцам принимать более обоснованные решения. Например, данные о ценах на жилье в различных районах могут быть использованы для прогноза ценовых колебаний, что крайне важно для инвесторов.

В-третьих, использование методов машинного обучения и аналитики на основе собранных данных может значительно повысить эффективность маркетинговых стратегий с точки зрения как продавцов, так и агентов по недвижимости. Это позволяет создавать индивидуальные предложения для клиентов и ориентироваться на более релевантные сегменты рынка.

1. Цели и задачи работы

В цели данной работы входит создание рабочего скрипта для парсинга данных с сайта недвижимости, скрейпинг данных и анализ полученных результатов в численном и графическом виде.

Поэтому в числе задач:

1. Изучение существующих методов парсинга
2. Выбор наиболее подходящего инструмента для парсинга
3. Разработка скрипта для парсинга
4. Сбор и обработка полученных результатов
5. Анализ собранной информации
6. Оценка полученных результатов
7. Объект и предмет исследования

Объектом исследования в данной работе является сайт недвижимости, содержащий информацию об объявлениях о продаже первичного жилья в городе Екатеринбург, как строящегося, так и недавно введенного в эксплуатацию.

Город Екатеринбург выбран не случайно. В этом городе одни из самых высоких темпов строительства в РФ, вне Москвы и МО, и ввода жилой недвижимости в эксплуатацию. Кроме того, присутствует личная заинтересованность в исследовании жилья в этом городе.

Также, предметом исследования является код сайта недвижимости. Т.е. изучение структуры сайта, изучение архитектуры, форматов данных.

Предмет исследования – процесс парсинга, т.е. изучение способов извлечения и обработки информации, получаемой с веб-сайта недвижимости.

В контексте работы над предметом исследования изучается комплекс программ и инструментов, используемых для парсинга, а также анализа полученных данных.

1. Структура работы

Цель данной дипломной работы – аналитика данных по рынку недвижимости в конкретном городе. Поэтому для полноценного погружения в тему, дипломный проект поделен на четыре части: введение, теоретическая часть, практическая часть и заключение.

Во вводной части рассмотрены актуальность данной темы объективно в текущих реалиях и, конкретно, для среднестатистического покупателя. Описываются цели и задачи данной работы, а, также, объекты и предметы исследования.

В теоретической части ведется полный экскурс по технологиям и инструментам, имеющим место в данной работе. Конкретнее, т.к. в работе рассматривается парсинг сайта недвижимости, анализ и визуализация полученных данных, то в теоретической части рассматривается каждый блок в отдельности (веб-скрейпинг/парсинг, аналитика данных, визуализация данных): дается определение каждому инструменту, описание существующих вариантов реализации инструментов и обоснование выбранных инструментов для данной работы.

В практической части, разбитой, также, на блоки, показан полный ход работы над проектом, от выбора рабочего сайта недвижимости и разработки приложения для парсинга, до аналитики и визуализации собранных в результате парсинга данных и кратких характеристик по ходу анализа данных. Конечно, с комментариями, описывающими мысли и трудности, с которыми приходилось сталкиваться в ходе работы над проектом.

В заключительной части обозначены итоги всей работы, даны основные рекомендации среднестатистическому покупателю и обозначены перспективы и возможные усовершенствования для данного исследования.

# Глава 1 Теоретическая часть

1. Обзор технологий веб-скрейпинга
   1. Понятие веб-скрейпинга и его применение

Веб-скрейпинг – это технология получения данных путем извлечения их со страниц веб-ресурсов.

Парсинг — это процесс анализа и извлечения данных из различных источников, таких как веб-страницы, документы или файлы и преобразования их в другой формат. Часто используется для обработки информации в автоматизированном режиме.

Применение парсинга:

- Извлечение данных с веб-сайтов: позволяет собирать информацию для анализа, например, цены товаров, новости и статьи.

- Анализ текстов: используется для исследований, обработки естественного языка и создания резюме текстов.

- Сбор статистики: помогает собирать данные для маркетинговых исследований или для оценки тенденций в определенной области.

- Автоматизация рутинных задач: упрощает процессы, которые требуют повторяющихся действий по сбору данных.

* 1. Этические аспекты парсинга

Сам по себе парсинг не является каким-либо нарушением или преступлением, прописанным законодательно. Однако, как и любой инструмент, он может быть использован и во благо, и в корыстных целях.

В подобных случаях, парсинг, как метод извлечения данных из различных источников, вызывает множество этических вопросов, особенно когда речь идет о веб-сайтах с конфиденциальной информацией.

1. Соблюдение правил использования:

- Условия использования сайта: Многие веб-сайты имеют «Условия использования», которые могут содержать оговорки об ограничении автоматизированного доступа к их содержимому. Парсинг сайта, который явно запрещает это в своих условиях, может рассматриваться как нарушение согласия владельца сайта.

- Файл robots.txt: Этот файл используется для указания, какие части сайта могут быть проиндексированы и доступны для парсинга. Несмотря на то, что соблюдение robots.txt не является юридическим обязательством, оно считается хорошей практикой.

2. Конфиденциальность:

- Личная информация: парсинг данных, содержащих личную информацию (например, адреса электронной почты, номера телефонов и другие идентифицирующие данные), без согласия пользователей является нарушением их конфиденциальности и может привести к правовым последствиям.

- законы о защите данных: в странах с жесткими законами о защите данных, такими как Евросоюз и прочие европейские страны, парсинг может привести к юридическим последствиям, если он противоречит принципам сбора и обработки данных.

3. Этическое использование данных:

- Важно намерение выполнения парсинга. Например, использование собранных данных для научных исследований, улучшения услуг или маркетинговых анализов может быть оправданным, в то время как использование для спама или мошенничества явно неэтично.

- Даже если данные собраны легальным способом, их использование может быть неэтичным, если оно намеренно вводит кого-либо в заблуждение или ухудшает чье-либо социальное положение.

4. Проблема нагрузки на сервер:

- Некоторые парсеры могут вызывать высокую нагрузку на сервер, особенно если они работают слишком агрессивно или запрашивают страницы слишком быстро. Это может привести к замедлению работы сайта или даже его временной недоступности, что делает такие действия неэтичными.

5. Запрет в условиях использования:

- Многие популярные сайты (такие как Facebook, LinkedIn, и другие) явно запрещают парсинг и автоматизированный доступ к своим данным. Нарушение этих правил может привести к блокировке доступа к сайту или даже судебным искам.

6. Технические меры защиты:

- Сайты могут применять различные технологии для защиты от парсинга: блокировка IP-адресов, использование CAPTCHA или динамическое изменение структуры контента для усложнения доступа.

7. Лицензионные ограничения:

- На некоторые данные могут распространяться лицензионные соглашения, которые ограничивают или запрещают их использование без разрешения владельца. Например, данные из научных публикаций могут находиться под юридической защитой.

* 1. Инструменты веб-скрейпинга: сравнение популярных библиотек

Для парсинга данных с веб-сайтов существует достаточно много инструментов, но остановимся на четырех наиболее популярных: BeautifulSoup, lxml, Scrapy и Selenium.

1. BeautifulSoup

- это библиотека Python для парсинга HTML и XML документов. BeautifulSoup предоставляет простые методы навигации, поиска и модификации дерева парсинга.

Среди его преимуществ –

- Простота использования, что очень хорошо для новичков.

- Хорошо обрабатывает "грязный" HTML, исправляя ошибки в разметке.

- Легко встраивается в другие библиотеки, такие как requests для получения HTML.

Однако, из его достоинств вытекают и недостатки:

- Не подходит для сложных запросов и веб-страниц высокой вложенности.

- Не подходит для больших объемов данных из-за низкой производительности.

- Не поддерживает асинхронность, что может замедлить процесс с большим количеством запросов.

2. lxml

- это более мощная библиотека, используемая для парсинга HTML и XML в Python, которая обеспечивает высокую производительность и поддержку расширенных возможностей.

Преимущества:

- Быстрее, чем BeautifulSoup, особенно при работе с большими документами.

- Поддерживает XPath, что облегчает извлечение данных из сложных структур.

- Может работать с валидными HTML и XML.

Недостатки:

- Более сложный в использовании по сравнению с BeautifulSoup.

- Иногда требует установки дополнительных зависимостей, что может вызвать проблемы при настройке.

3. Scrapy

- это фреймворк для веб-скрейпинга, который позволяет создавать пауков для автоматизированного извлечения данных из веб-сайтов. Использует асинхронное программирование для повышения производительности.

Преимущества:

- Подходит для сбора больших объемов данных благодаря встроенной асинхронной обработке.

- Имеет мощные средства для обработки различных форматов данных (JSON, CSV и др.).

- Встроенные механизмы для работы с очередями, фильтрации и управления сессиями.

Недостатки:

- Более сложен в освоении для новичков.

- Может быть избыточным для простых задач парсинга.

4. Selenium

- это инструмент, используемый для автоматизации браузеров. Часто применяется для парсинга динамически загружаемых веб-страниц (например, с JavaScript).

Преимущества:

- Способен работать с веб-приложениями, которые используют JavaScript, поскольку управляет реальным браузером.

- Позволяет взаимодействовать со страницами: щелчки, скроллинг, ввод данных и т. д.

Недостатки:

- Медленнее, чем другие инструменты, так как он запускает реальный браузер.

- Требует больше ресурсов (памяти, процессора) для работы.

- Сложнее настраивать и использовать по сравнению с библиотеками для парсинга.

1. Изучение библиотеки BeautifulSoup
   1. Основные возможности

BeautifulSoup — это библиотека для парсинга HTML и XML документов в Python, которая позволяет пользователям извлекать и манипулировать данными из веб-страниц. При работе с HTML или XML документами, BeautifulSoup берет эти текстовые данные и создает дерево элементов (DOM-структуру). Это дерево позволяет пользователю легко взаимодействовать через парсеры (такие как lxml и html.parser) с различными частями документа через функции поиска по тегам, id, текстовому содержимому между тегами и т.п.

К примеру:

- find(tag, attrs): находит первый элемент с заданным тегом и атрибутами.

- find\_all(tag, attrs): находит все элементы с заданным тегом и атрибутами.

Чтение полученного HTML-кода легче воспринимать, если применить к коду метод prettyfy(), который форматирует документ с соответствующими отступами.

* 1. Установка

Установка библиотеки BeautifulSoup производится стандартными действиями в терминале через pip install beautifulsoup4.

Дальнейшая работа происходит в любом интерпретаторе через импорт библиотеки BeautifulSoup, соответствующего парсера для работы и библиотеки requests.

from bs4 import BeautifulSoup

from lxml import html

import requests

1. Анализ данных о недвижимости
   1. Изучение сайтов недвижимости

В настоящее время сайты недвижимости представляют из себя огромные базы данных, содержащие всю информацию о квартире (цена, количество комнат, квадратные метры, этажность и т.п.), о доме (район, год постройки/сдачи в эксплуатацию), улице (наличие рядом метро, сад, школ и т.п.) и, даже, районе.

Поэтому изучение данных на подобных сайтах предоставляет широкие возможности для изучения.

Наибольшие веса в данной нише заняли такие сайты недвижимости, как Циан, Дом-клик, Авито, Яндекс-недвижимость.

* 1. Типовые данные, собираемые с сайтов недвижимости для последующей оценки и анализа

Для целей данной работы наибольший интерес представляют такие данные, как:

- количество квадратных метров жилплощади, м2

- цена, млн.руб.

- номер этажа

- тип квартиры (студия, 1-но, 2-ух, 3-ех комнатная)

Эти данные имеют наибольшее значение при выборе жилья при прочих равных условиях. Кроме того, такие данные проще анализировать, интерпретировать и выстраивать зависимости.

* 1. Методы и инструменты для анализа данных

Для парсинга сайтов недвижимости нет какого-то определенного шаблона, нет наиболее эффективного способа соскрейпить информацию, нет наиболее удачной библиотеки для парсинга.

В зависимости от сложности и вложенности сайта недвижимости, наиболее целесообразно применять разные библиотеки. Как уже указывал выше, при рассмотрении инструментов-библиотек для парсинга веб-страниц, для простых сайтов подойдут такие простые библиотеки, как BeautifulSoup или lxml. При более сложных и вложенных сайтах, или при необходимости более гибкой автоматизации процессов, можно использовать Scrapy. А для еще большей имитации реального поведения пользователя, лучше использовать Selenium.

1. Визуализация данных с помощью библиотек Matplotlib и Seaborn
   1. Основы визуализации данных

Визуализация данных – в данном случае, в сфере недвижимости - играет важную роль в принятии обоснованных решений. Существует множество факторов, таких как местоположение, цена, характеристики объекта и рыночные тренды, которые практически невозможно оценить и интерпретировать только по численным данным, не имея графического представления. Эффективная визуализация позволяет не только упростить восприятие этих факторов, но и выявить скрытые зависимости и тренды.

Через правильно подобранный график можно быстро и точно оценить тенденцию рынка, сравнить цены на аналогичные объекты, определить потенциальные инвестиционные возможности, проследить изменения в ценах и динамику спроса и предложения на рынке.

* 1. Сравнение библиотек Matplotlib и Seaborn

Для визуализации данных и презентации работ клиентам, заказчикам, инвесторам и, просто, коллегам внутри компании, чаще всего пользуются отдельными BI-инструментами, такими как PowerBI, Superset, Tableau. Данные инструменты позволяют настраивать любой тип графиков под любые нужды и пожелания, от цветовой палитры до создания уникальных зависимостей между дата-сетами.

В то же время, для визуализации данных в ходе рабочего процесса, т.е. прямо в интерпретаторе, имеются библиотеки, позволяющие сделать достаточно качественное графическое представление. Такие библиотеки применяются на этапе разведочного анализа, когда необходимо оценить проделанную работу по загрузке и очистке данных, быстро оценить зависимости и провести сравнение представленной информации.

К наиболее популярным библиотекам визуализации данных для python относятся Matplotlib и Seaborn.

Для моей дипломной работы наибольший интерес вызывает анализ данных в графическом виде через эти библиотеки, т.к., по моему мнению, одним из основных качеств продуктового аналитика является скорость принятия решений о ходе своей работы. Поэтому оценка данных "здесь и сейчас" очень важна, а использование соответствующих библиотек этому очень помогает.

Рассмотрим основные различия и назначение этих двух библиотек:

Matplotlib – это основная библиотека для визуализации данных в python. Она предоставляет широкий спектр функций и настроек для создания различных типов графиков, линий, точек и т.п.

С помощью этой библиотеки можно создавать сложные и настраиваемые графики с высокой степенью контроля отображения. Отсюда же вытекает и недостаток данной библиотеки – сложность для новичков и необходимость создания большего кода, чем следующая библиотека.

Seaborn – библиотека для визуализации, построенная на основе Matplotlib, но ориентирована на более простое отображение и настройку графиков и удобство работы с кодом. Как и в прошлом примере, достоинство библиотеки обнажает скрытые за ним недостатки – меньше возможностей для кастомизации, настройки отображения визуальной части графиков, стили и цветовые палитры "из коробки".

* 1. Типы графиков, применяемых для анализа данных о недвижимости

Для целей аналитики в данной дипломной работе я использовал четыре вида графиков, наиболее явно и наглядно отражающих результаты аналитических изысканий.

Эти графики plt.plot(), sns.boxplot(), sns.catplot(), sns.lmplot из библиотек Matplotlib и Seaborn.

Рассмотрим подробнее каждый график, его основное применение и достоинства перед другими типами графиков.

1. plt.plot()

Функция plt.plot() из библиотеки Matplotlib предназначена для создания двумерных линейных графиков. Этот базовый график позволяет визуализировать зависимость между двумя переменными. Далее, в графической части работы, он играет роль графика рассеяния, по которому можно увидеть места наибольшего скопления определенных данных, относящихся к каждому виду.

2. sns.boxplot()

Функция sns.boxplot() из библиотеки Seaborn используется для построения графиков коробка-с-усами (боксплотов), которые визуализируют распределение данных, показывая медиану, квартиль и выбросы, и всё это на одном графике. Пожалуй, один из самых информативных графиком, применяемых в анализе данных недвижимости в данной работе.

3. sns.catplot()

Функция sns.catplot() используется для создания различных категориальных графиков, таких как столбчатые графики и графики с уровнями. Этот метод является весьма универсальным и подходит для представления и сравнения подгрупп данных. В данной работе, график идеально подошел для демонстрации различий разных типов квартир (студия, 1-, 2-, 3-х комнатная) по цене и количеству квартир, предлагаемых к продаже.

4. sns.lmplot()

Функция sns.lmplot() используется для построения графиков линейной регрессии. Он позволяет визуализировать зависимость между двумя переменными и накладывает регрессионную линию на данные.

Применение графика в данной работе позволяет увидеть тенденцию изменчивости цен в зависимости от метража квартиры.

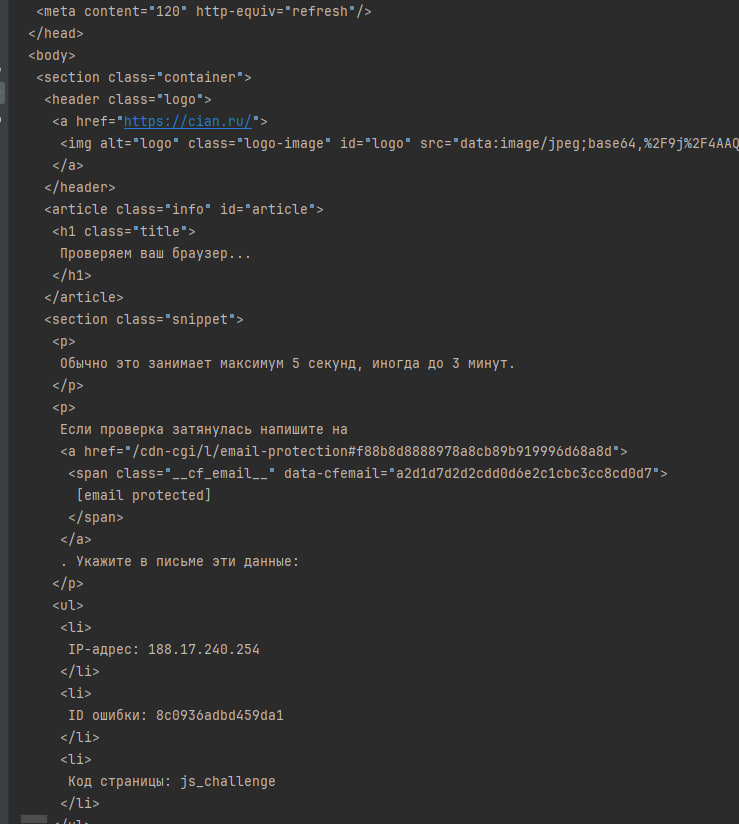
# Глава 2 Практическая часть

1. Выбор сайта и сред разработки

Т.к. вся работа состоит из четырех логически обособленных частей, то я для каждого этапа я выбрал 4 разных среды разработки, наиболее подходящих для своего случая.

Первая часть – сам сайт, данные с которого я буду парсить.

Изучив четырех гигантов сферы продаж недвижимости, я наткнулся на стену, под названием нейросети, отслеживающие парсеры по TLS-отпечаткам (пример ответа на request-запрос для сайта cian.ru из-за проверки на бота сайтом).



Не имея возможности через эту стену перебраться (на данном этапе своего развития в качестве аналитика данных), я решил проблему другим путем – сменив сайт-гигант на сайт-карлик с меньшим количеством представленных к продаже квартир, но с более простой архитектурой сайта и менее жесткой службой безопасности. Выбор пал на сайт atomstroy.net.

Вторая часть – разработка скрипта парсера.

Наиболее популярные для python разработчиков среды программирования – PyCharm и VisualStudio, JupyterNotebook и google.collab. Первые две – это полноценные IDE, а вторые – так называемые ноутбуки.

Для написания кода я выбрал PyCharm, как наиболее близкую мне IDE. В данном случае ноутбук не подойдет, т.к. скрипт должен быть воспроизведен одним действием, в то время как в ноутбуке каждая часть кода воспроизводится в отдельном окне и может быть запущена без привязки к прошлому коду.

Третья часть – оценка и корректировка/очистка полученных данных в файле .csv.

Для проверки и очистки данных использовалась стандартная программа Microsoft Excel. В отличие от кода на python, в Excel для очистки данных не требуется написание кода, поэтому после пары действий, все данные были проверены на достаточность и отсутствие пропусков, а также все запятые заменены на точки. Данная программа позволяет выполнять подобные действия достаточно быстро, в отличии от написания кода. Однако автоматизированную очистку данных при последующих парсингах настроить нельзя.

Четвертая часть – анализ очищенного и подготовленного датасета и построение графиков.

Как было указано выше, для данной цели наиболее эффективно использовать ноутбук. В данной работе был использован Jupyter Notebook.

1. Разработка приложения для парсинга данных с сайта недвижимости

# импортируем библиотеки, необходимые для парсинга  
import re # модуль для записи регулярных выражений  
import requests # пакет для отправки запроса на сайт   
from bs4 import BeautifulSoup # пакет для парсинга сайта  
import pandas as pd # пакет для работы с данными в DataFrame виде  
  
# создаем пустой список. Будем наполнять его данными слой за слоем  
rows = []  
# т.к. сайт достаточно простой, то для обхода по каждой странице достаточно менять значение страницы по порядку,   
# начиная с 1 и заканчивая 19. Решил сделать цикл for-in. Для простой задачи - простое решение

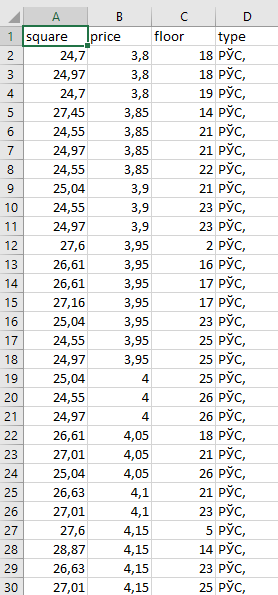
for i in range(1, 19):  
 url = f'https://www.atomstroy.net/kvartiry?page={i}' # назначаем адрес сайта для url через f-строку. Значение i  
 # меняется при каждом обходе по циклу  
 # далее прописываем в переменную ответа функцию с url и User-Agent, для большей имитации реального запроса через   
 # браузер человеком  
 response = requests.get(url, headers={  
 'User-Agent' : 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) '  
 'Chrome/127.0.0.0 Safari/537.36'})  
 # при каждом новом цикле for-in, парсер скрейпит html-страницу и парсит ее через html.parser  
 soup = BeautifulSoup(response.content, "html.parser")  
 # print(soup.prettify()) - # это я смотрел красивый html-код в выводе IDE

# создаю цикл for-in для записи в список rows спарсенных данных в формате словаря  
for row in soup.find\_all('li', ('class', 'catalog-list\_\_item catalog-list-item')):  
 rows.append({  
 'square': row.find('span', ('class', 'color color--red')).text[:-3],  
 'price': row.find(string=re.compile("млн")).text[:-4],  
 'floor': row.find('span', ('class', 'catalog-list-item\_\_text')).text[:-5],  
 'type': row.find(string=['Ст', '1к', '2к', '3к', '4+'])  
 })

# создаю из списка rows DataFrame для более простого преобразования данных в табличном виде  
df = pd.DataFrame(rows)  
# и сохраняю все это в формате .csv с разделителем, дефолтным для Excel  
df.to\_csv(r'./apt\_data-frame.csv', sep=';', encoding='utf-8', index=False, header=True)

1. Сбор и обработка данных

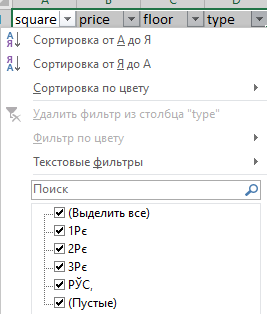
После получения файла "apt\_data-frame.csv", получился достаточно чистый файл, для дальнейшей работы в программе Excel.



Но обнаружилось несколько проблем:

Первая проблема - в строке типа квартиры оказались пустые ячейки, которых оказалось всего две из 539. Поэтому проблема решилась удалением этих двух нестандартных ячеек.

Вторая проблема – корявое отображение самого типа квартир. Открыв полученный датасет в Jupyter Notebook всё оказалось в нормальном состоянии, поэтому тратить временные ресурсы на решение несущественной проблемы оказалось нерационально.



И третья – самая критичная проблема – запятые в числах. На этапе парсинга и обработки данных в Excel, эти значения подозрений не вызывали, даже наоборот, в Excel обрабатывались как надо. Однако при работе в Jupyter Notebook, данные этих столбцов определялись как объектные типы данных, и при арифметических операциях появлялись ошибки.

Поэтому через функцию "ПОДСТАВИТЬ(ячейка;",";".")" за несколько секунд получилось откорректировать все данные под нужный формат и уже теперь в Jupyter Notebook все данные определялись корректно и можно было приступать к анализу полученных данных.

1. Анализ собранных данных

Для целей анализа полученных данных используется инструмент типа ноутбук, а именно, программа Jupyter Notebook. Инструменты такого типа наиболее удобны как раз для целей разведочного анализа в виде таблиц и графиков, т.к. видеть результат обработки данных можно практически после каждой строки кода.

В начале любого анализа в среде разработки на языке python, необходимо импортировать соответствующие библиотеки, которые будут использованы для последующего использование в коде.

Для загрузки и анализа данных в табличной форме нужна библиотека Pandas. Pandas — это одна из самых популярных и мощных open-source библиотек для анализа и обработки данных на языке программирования Python. Она обеспечивает удобные структуры данных и инструменты для работы с ними, позволяя проводить анализ и манипуляции с данными на высоком уровне.

Эта библиотека импортируется через вызов:

import pandas as pd

, где pd – общепринятое сокращения для обращения к библиотеке pandas, при написании кода.

Также, для работы с данными, понадобится библиотека Numpy.

NumPy (Numerical Python) — это библиотека для языка программирования Python, которая предоставляет мощные инструменты для работы с многомерными массивами и матрицами, а также набор математических функций для выполнения операций над этими данными. Она является основой для многих других библиотек в научных вычислениях и анализе данных, таких как Pandas, SciPy, Matplotlib и других.

Эта библиотека импортируется через вызов:

import numpy as np

Для графического анализа и отображения полученных результатов будут использоваться такие графические библиотеки, как Matplotlib и Seaborn.

import matplotlib.pyplot as plt

from matplotlib import rcParams

import seaborn as sns

И так, рассмотрим имеющийся датасет, импортировав нужные нам столбцы:

df\_raw = pd.read\_csv('apt\_data-frame.csv', sep=';')

df = df\_raw[['square', 'price', 'floor', 'type']]

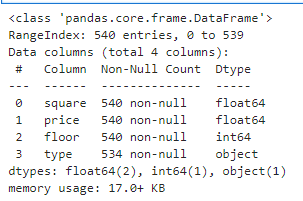
Чтобы понять, в каком виде появляются данные, выведем первые 10 строк:

df.head(10)



Проверим качество данных – тип и количество ненулевых значений:

df.info()



Краткая сводка по всем числовым колонкам:

df.describe()



Рассмотрим такую же сводку, но применительно к квартирам каждого типа, применив к датафрейму селектор по типу:

studio = df[df['type']=='Ст']

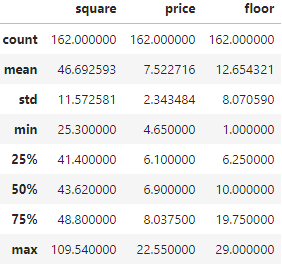
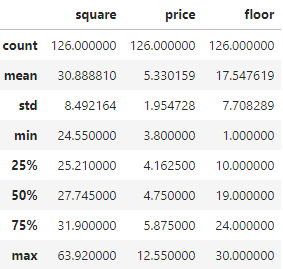
one\_room = df[df['type']=='1к']

two\_room = df[df['type']=='2к']

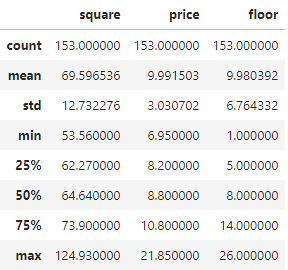
three\_room = df[df['type']=='3к']

И теперь сделаем сводки:

studio.describe() one\_room.describe()

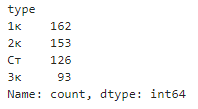


two\_room.describe() three\_room.describe()

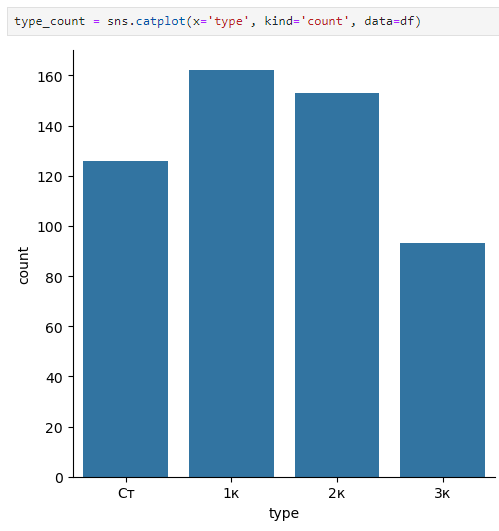


Посмотрим, сколько квартир каждого типа имеется в датасете:

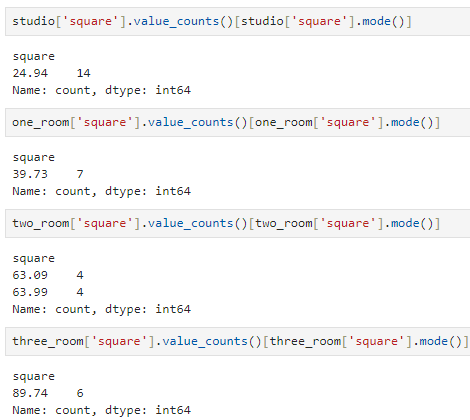
df["type"].value\_counts()



На графике catplot та же информация выглядит более ясно и наглядно:



Найдем моду (наиболее часто повторяющееся значение метража предлагаемых к продаже квартир разного типа):



И ту же моду к значению цены каждого типа квартир:



Кратко сформулирую наблюдения, после полученного анализа:

1. Студии и однокомнатные квартиры, при равной площади отличаются в цене на 1 .. 2 млн.руб. (1-комнатные дороже)
2. В среднем, однокомнатные квартиры дороже 2-х и 3-х комнатных при меньшей площади.
3. Несмотря на то, что социальная норма жилья для одного человека:

- живущего в одиночку – 33 м2

- живущего в составе семьи из 2 человек – 21 м2

- живущего в составе семьи из 3 и более человек - 18 м2

при минимальных значениях, студии и 1-но комнатные квартиры не отвечают этим требованиям.

1. В среднем, студии, 1-но, 2-х и 3-х комнатные квартиры отличаются друг от друга, с увеличением площади на 20 м2, увеличением цены на 2 млн.руб.
2. Наиболее часто повторяющаяся площадь студии находится у ее минимального значения.

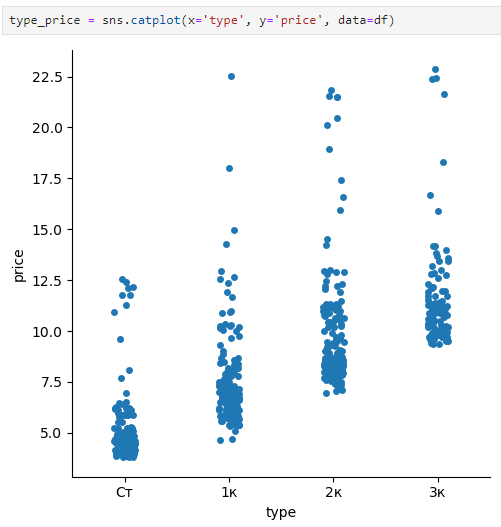
Однокомнатные квартиры предлагаются наиболее часто с количеством квадратных метров около 40 м2, что примерно равняется медианному значению.

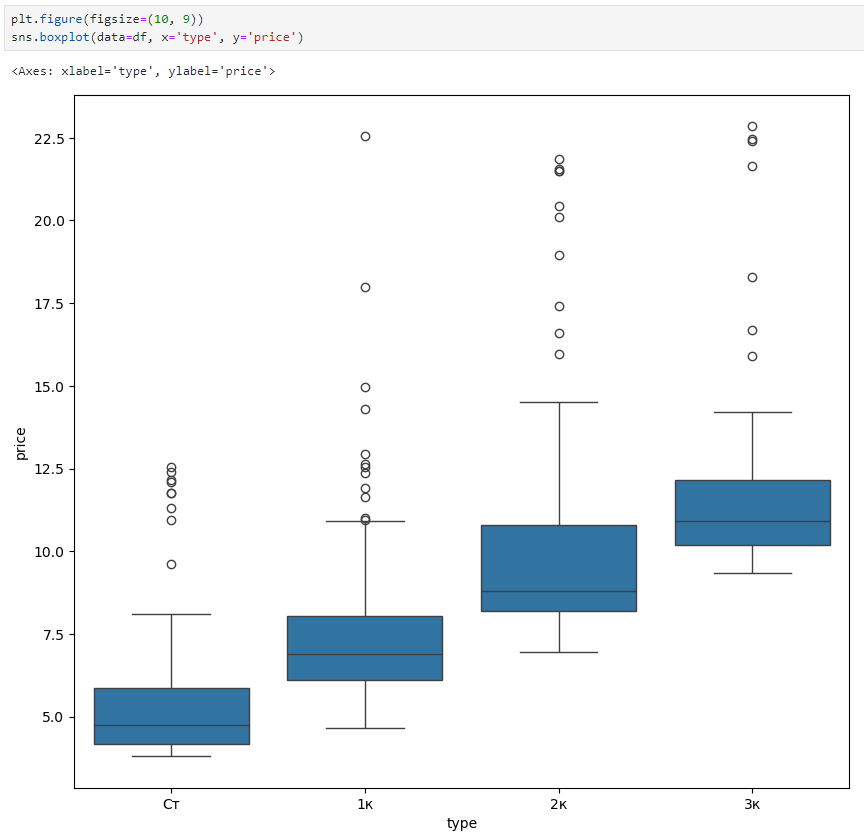
То, что сказано про 1-но комнатные квартиры, справедливо и для 2-х и 3-х комнатных.

1. Визуализация результатов анализа

Теперь рассмотрим графическое представление анализа датасета:

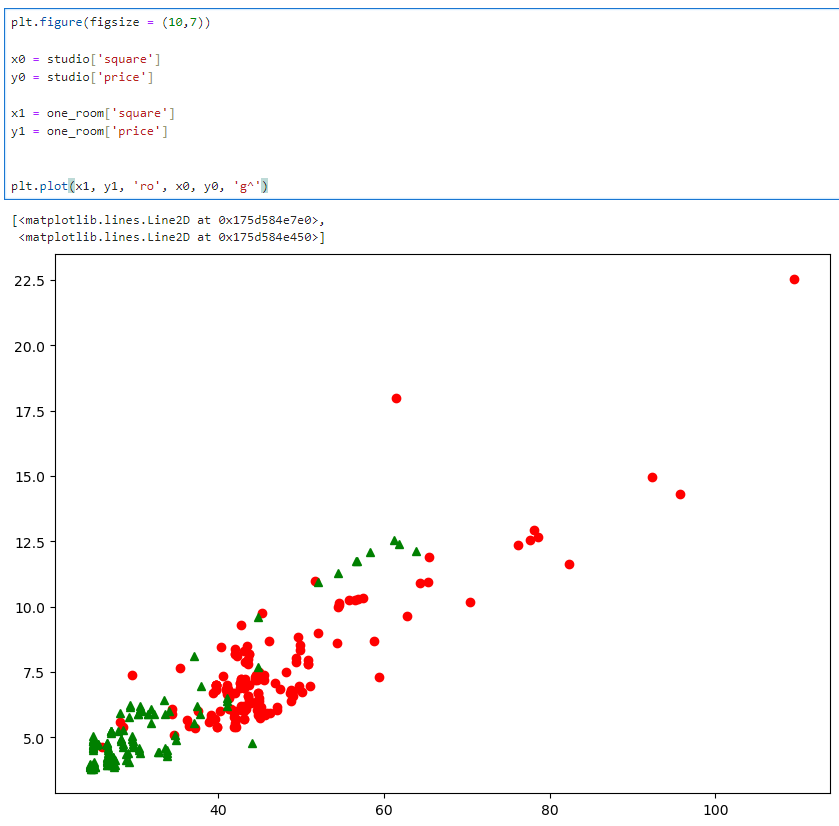
Оценим распределение квартир каждого типа в ценовом диапазоне через графики catplot и boxplot библиотеки Seaborn.



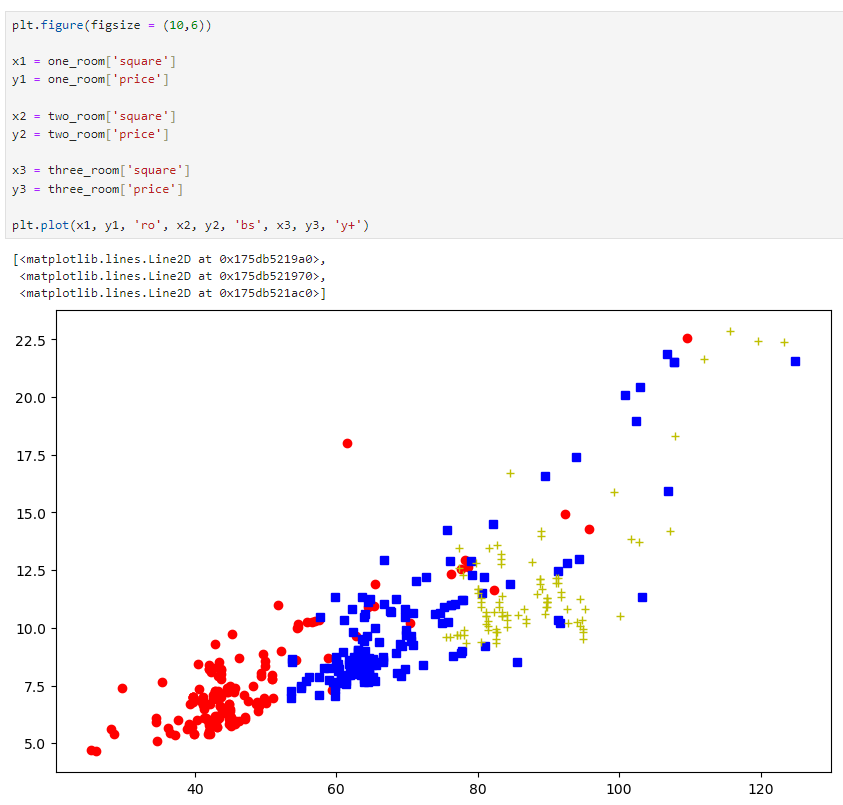


Как видно из графика, выбросов достаточно много, и при цене свыше 15 млн.руб. цена практически не зависит от типа квартиры, но основная часть квартир всех типов находится в пределах ценового диапазона от 3 млн.руб. до 12 млн.руб. Что, в принципе, было видно ранее на этапе численной оценки значений.

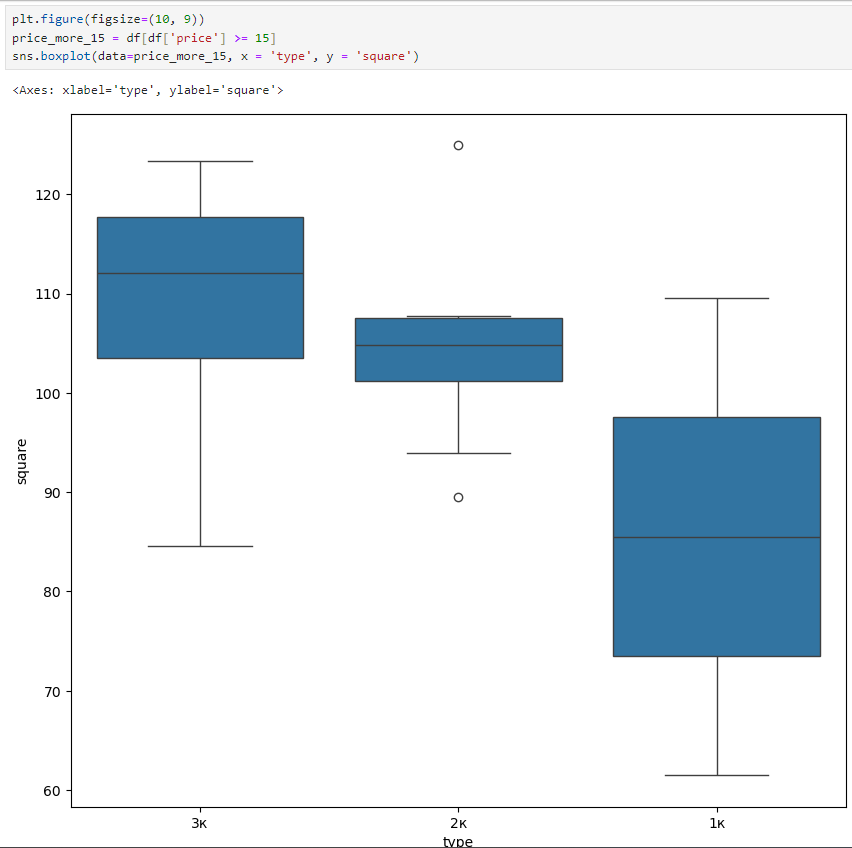
Сравним разброс значений студий и однокомнатных квартир в координатах зависимости цен от площади через график plot библиотеки Matplotlib:



И, сделаем то же для сравнения однокомнатных, двухкомнатных и трехкомнатных квартир:

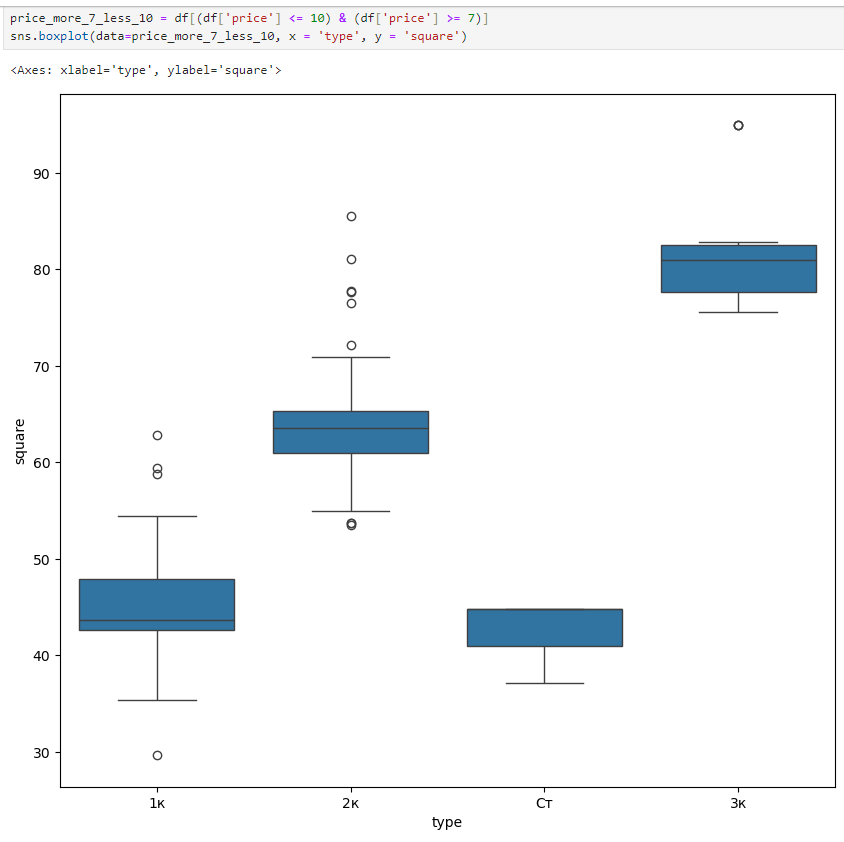


Если рассматривать только квартиры в ценовом диапазоне от 15 млн.руб., то распределение по метражу будет следующим:

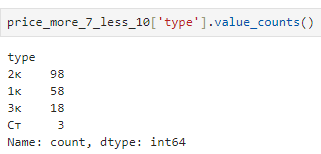


Что также подтверждает выводы сделанные ранее – при повышении цены, однокомнатные квартиры при сопоставимой с 2-х и 3-х комнатными квартирами имеют меньший метраж.

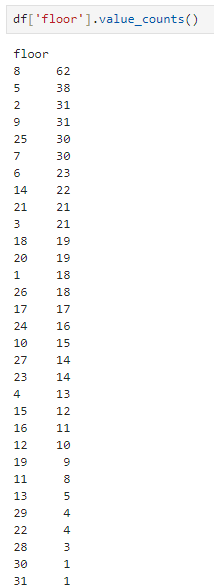
Предлагаю рассмотреть, на какой метраж какого типа квартир может рассчитывать покупатель, имея возможность купить квартиру за цену от 7 млн.руб. до 10 млн.руб. График представлен ниже:



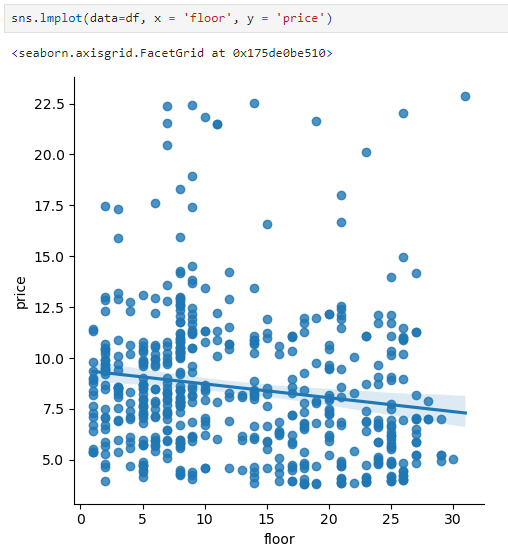
И каков объем предложений по типу квартир в данном ценовом диапазоне:



Рассмотрим количество этажей наиболее часто предлагаемых к продаже квартир:



И то же на графике lmplot:



Как численные данные, так и график показывают, что количество предлагаемых к продаже квартир на более верхних этажах, меньше, чем количество квартир, продаваемых на нижних этажах.

Отсюда можно сделать вывод, что квартиры на верхних этажах раскупают быстрее, чем на нижних, поэтому в активной продаже остается больше квартир на нижних этажах.

При этом, как видно по графику зависимости цены от этажа – определенной зависимости нет.

# Заключение

## Подведение итогов работы

В результате текущей дипломной работы выбран сайт, содержащий необходимую для аналитики информацию, написан скрипт по парсингу данных с выбранного сайта и проведена аналитика полученного датасета.

В ходе работы описаны инструменты для парсинга и аналитики информации, обоснован выбор конкретных инструментов.

1. Основные выводы и рекомендации

В этом разделе я представлю основные выводы и рекомендации касательно результатов аналитики датасета. Выводы и рекомендации по самому процессу исследования находятся в следующем разделе.

И так, по результатам анализа определились некоторые интересные закономерности, которые хотелось бы подсветить:

1. Студии, как и ожидалось – наиболее дешевый вариант из всех типов квартир как по минимальной цене, так и по максимальной. При этих своих характеристиках, они, также, имеют наименьшую квадратуру как по минимальному, так и по максимальному значению.

Максимальные значения по цене и квадратуре у квартир-студий, примерно, в два раза меньше максимальных значений у остальных типов квартир.

1. Однокомнатные, двухкомнатные и трехкомнатные квартиры имеют практически равные характеристики, начиная с диапазона цен в 10 млн. руб.

При этом, при равных площадях, цены однокомнатных квартир выше, чем у 2-х и 3-х комнатных.

1. В ценовом диапазоне 7 .. 10 млн.руб. больше всего к продаже представлено 2-х комнатных квартир (98 шт). Далее идут 1-о комнатные (58 шт). Меньше всего в данном ценовом диапазоне 3-х комнатных квартир (18 шт) и студий (3 шт).

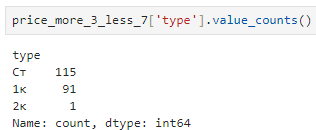
Как специалист в сфере девелопмента, я знаю, что наибольшее количество типов квартир при проектировании общедоступных многоквартирных домов (МКД), отдается однокомнатным и двухкомнатным квартирам. Студии и трехкомнатные квартиры, в среднем, закладываются в меньшем объеме в 2 и более раз.

Однокомнатные, двухкомнатные и трехкомнатные квартиры большего метража закладываются в более элитных районах, более нестандартных МКД.

Данная тенденция и прослеживается при анализе доступных квартир в диапазоне цен 7 .. 10 млн.руб.

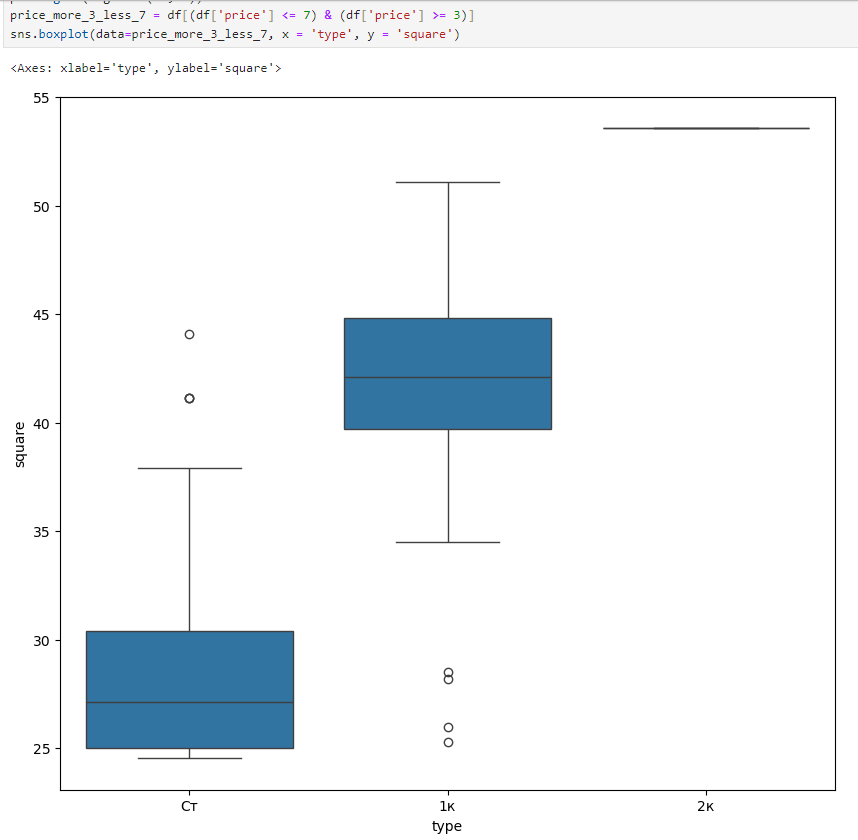
1. В качестве бонуса, хотелось бы рассмотреть, на что можно рассчитывать при финансах от 3 млн.руб. до 7 млн.руб.

Во-первых, объем предложений при таком ценовом диапазоне:



Что означает, что на 2-х и 3-х комнатные квартиры не стоит и надеяться.

Логично, что студий, с их мизерным метражом и низкими, по сравнению с квартирами остальных типов, ценами, намного больше 1-о комнатных квартир. Но, посмотрим на квадратуру предлагаемых квартир:



Видно, что наибольшие значение квадратуры приходятся на диапазон 25 .. 30 м2 у студий и у однокомнатных квартир.

Конечно, есть редкие случаи, где у студий количество квадратных метров находится в диапазоне 40 .. 45 м2 и однокомнатных квартир с 25 .. 30 м2, но это, скорее, исключение из правил, чем закономерность.

Рекомендации, которые можно вывести как следствие из полученной аналитики, опираясь на социальные нормы метража на человека:

1. Для проживания одного человека в квартире с площадью от 33 м2 потребуется примерно 5,2 млн.руб.

Но, считая, что на одного человека допустимо принимать 18 м2, то можно найти студию за 3 млн.руб., либо однокомнатную квартиру за 5 млн.руб. с 30 м2. Такой вариант подойдет скорее для сдачи в аренду, либо для покупки для проживания одному человеку, не имеющему в планах заводить семью.

1. Для проживания семьи из двух человек (в расчете 42 м2 на обоих) можно найти однокомнатную квартиру за 5,5 млн.руб.
2. Для семьи из трех человек (в расчете 58 м2 на троих - 18 м2 на одного) можно найти однокомнатную и двухкомнатную квартиру за 7,5 млн.руб.
3. Максимальный вариант трехкомнатных квартир при потолке в 10 млн.руб. – это квартира с 82 м2. Причем минимальная стоимость трехкомнатных квартир начинается с отметки в 9,4 млн.руб. за 75 м2.

Подытоживая вышеперечисленное, в зависимости от целей покупки квартиры, можно найти подходящий вариант уже от 3 млн.руб. для сдачи в аренду либо покупки, в качестве начального капитала для студента или человека, только начавшего рабочий путь. Либо придется заплатить от 5,5 млн.руб. для покупки жилья для молодой семьи.

1. Перспективы и совершенствования исследования

Хочется начать с того, что т.к. для парсинга был выбран сайт с низкой безопасной защищенностью, то это сайт с малым количеством вариантов предложений квартир к продаже (суммарно 540 квартир), что достаточно хорошо и репрезентативно для выборки для выбранного города, но не так хорошо, как 25 тысяч вариантов квартир на сайте ЦИАН.

Так что для еще большего количества вариантов и возможных предложений, а также, возможно, удачных выбросов, удовлетворяющих требованиям к цене и квадратуре, текущий сайт недвижимости будет заменен на более популярный сайт.

Для этого необходимо изучить методы и инструменты для обхода ИИ-ботов и tls-отпечатков.

Настроив, через Selenium возможность заполнять поле "город/населенный пункт" на сайте, можно сделать поиск не только по одному дефолтному городу, заложенному в код изначально, но и подстроив парсер под себя.

Библиотека BeautifulSoup хорошо зарекомендовала себя для поиска данных, как по тегам, так и по имеющейся внутри тегов текстовой информации, поэтому менять сам принцип работы парсера – не имеет смысла.

Подытоживая данный раздел и всю дипломную работу – работа выполнена на уровне, достаточном для получения полезной и достаточно качественной аналитики по жилой недвижимости в городе Екатеринбург. Парсер выполняет свою работу по сбору данных с конкретного сайта. Собранные данные достаточно чистые для минимальной обработки и использования далее для аналитики. Изучение данных через Pandas и графической интерпретации через Matplotlib и Seaborn дает широкое представление полученных данных и рассмотрение их с разных сторон для различных ситуаций и требований.

Однако, остановиться на этом этапе – большая ошибка. На текущем этапе данным проектом уже заинтересовались несколько человек, работающих в сфере недвижимости. Поэтому данному проекту необходимо докрутить функционал и, возможно, настроить отображение графиков уже на этапе обработки данных в Excel, либо перевести данные в BI-инструменты. В любом случае, разведочный анализ потеряет свою актуальность и использование ноутбука будет излишней.

# Список используемой литературы

1. <https://ru.wikipedia.org/>
2. <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>
3. <https://lxml.de/>
4. <https://selenium-python.readthedocs.io/>
5. <https://docs.scrapy.org/en/latest/>
6. <https://matplotlib.org/stable/index.html>
7. <https://seaborn.pydata.org/>