Очень важно для проведения такого рода анализов осуществить доскональное изучение доступных источников данных, провести полноценный анализ сферы коммерческой недвижимости и принципов оценки объектов. []

Для анализа рынка и построения в дальнейшем модели – необходимо собрать основную базу данных недвижимости. В организации, как правило, используется собственная база данных, однако в рамках конфиденциальности собственности компании возникает необходимость использовать данные с открытых источников, тем не менее, в самой компании регулярно практикуется сбор и использование открытых источников, в особенности для обновления базы данных и понимания функционирования рынка. Также, стоит отметить, в рамках работы будут использованы данные именно по Санкт-Петербургу и ближайшей части Ленинградской области, но это не отменяет возможностей использования данного подхода к любой местности, при наличии основных параметров.

На нынешнем рынке есть несколько крупных открытых баз недвижимости. В первую очередь это такие сайты как «Авито», «Циан», «Яндекс Недвижимость». Однако есть и другие, достаточно подробные информационные источники, однако они принадлежат определенным агентствам недвижимости, то есть подразумевают узконаправленность имеющихся данных и не предоставляют необходимый для данного рода работы объем и спектр объектов.

Из указанных источников необходимо выбрать один основной, соответствующий основным параметрам:

* Полнота данных – наличие максимума информации об объекте;
* Целостность и корректность данных;
* Отсутствие ложных данных;
* Наличие критически важных параметров: адрес, квадратура, стоимость.

Стоит рассмотреть конкретные примеры объектов на данных площадках. В таком случае будет возможность изучить возможный набор данных, который можно будет собрать с каждого объекта.

На данном рисунке видно пример объекта с сайта «Яндекс Недвижимость». Если изучить код страницы, а также визуальную составляющую – со страницы объявления можно извлечь следующую информацию:

* URL;
* Стоимость;
* Площадь;
* Стоимость за квадратный метр;
* Адрес;
* Этаж;
* Назначение (офис, склад, торговое);
* Фиктивные переменные, т.е. подробная информация;
* Агентство / Собственник;
* Расстояние до метро в минутах.

Далее рассмотрим пример страницы объявления Авито:

* Стоимость;
* Площадь;
* Стоимость за квадратный метр;
* Фиктивные переменные, т.е. подробная информация;
* Адрес;
* Этаж;
* Агентство / Собственник;

Как последний подробно-изученный источник для будущей базы данных – рассмотрим Циан.

* URL;
* Стоимость;
* Площадь;
* Стоимость за кв метр;
* Дом;
* Этаж;
* Назначение (офис, склад, торговое);
* Тип здания;
* Фиктивные переменные;
* Агентство;
* Расстояние до метро в минутах;

С учетом проведенного анализа, включая изучение корректности данных – было решено, что основным источником для данных стоит использовать Циан. Яндекс и Авито не так подробно изучают объявления и оставляют большое количество ложных объявлений, которые размещают некоторые агентства недвижимости для привлечения клиентов.

ААААААААААААААА

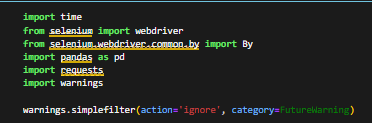
Для сбора данной информации придется обратиться к написанию программы-парсера. Парсер — это программа, сервис или скрипт, который собирает данные с указанных веб-ресурсов, обрабатывает их и выдает в нужном формате. Использование парсеров очень распространено в современном мире, во многом это один из основных способов получения информации из открытого доступа. Более того, в сфере недвижимости это цифровой фундамент для изучения рынка. Все современные агентства стараются постоянно следить за новыми данными и собирать максимум из них. Такой задачей и занимается один из отделов компании «Актив-Инвест».

После изучения основных источников на тему парсеров, их типов, принципов использования и основных библиотек на языке программирования python возникла необходимость также и в изучении основ написания сайтов – html, для понимания структуры страниц и данных на этих страницах. Проконсультировавшись со специалистами в «Актив-Инвест» – была проделана работа по написанию парсера специально для данного сайта, с целью собрать максимум возможной информации.

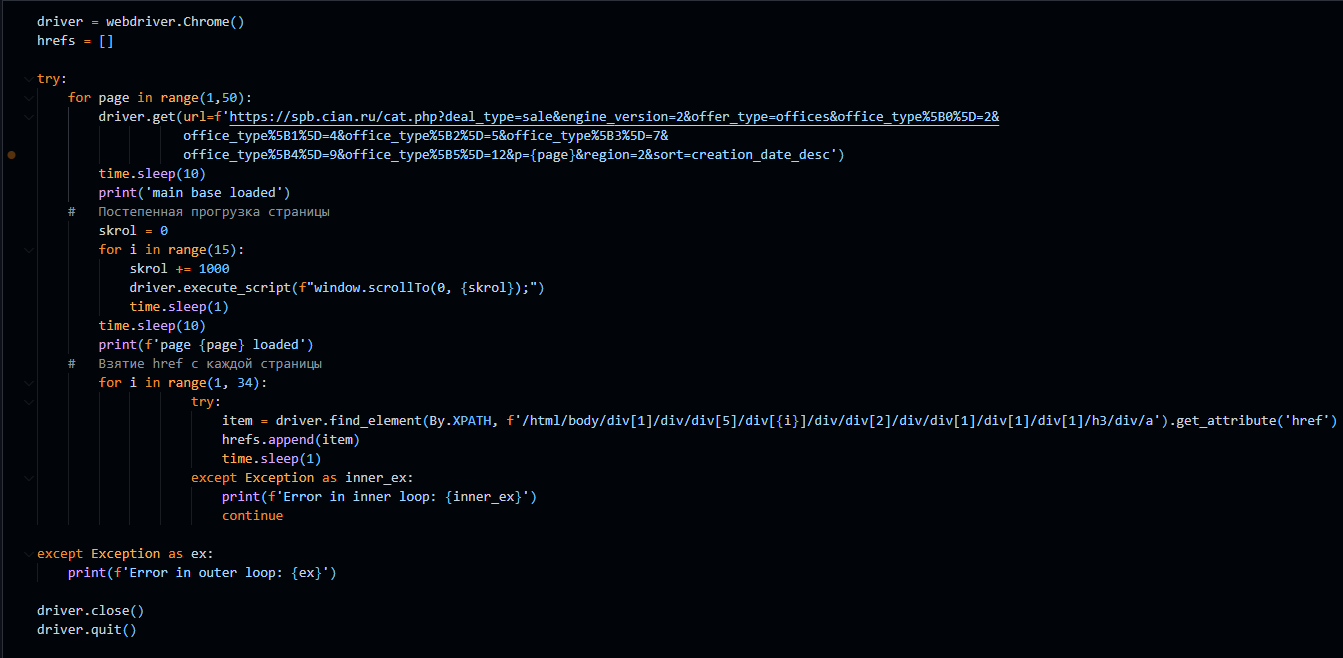
Для данной задачи были изучены основные библиотеки для парсинга на языке программирования python – “Beautifull4Soup” и “Selenium”. Используя основные API этих библиотек, были изучены возможности для сбора данных.

Выводом стало, что самым оптимальным и эффективным способом стоит считать использование библиотеки “Selenium”, во многом благодаря возможностям данной библиотеки более точно взаимодействовать с сайтами, используя имитацию нажатию кнопок, открытия ссылок и перехода от страницы к странице.

Для того, чтобы собрать все объекты, первоначально необходимо собрать все ссылки на них с предварительно настроенной фильтрацией сайта. Настроим среду для кода.



Вот итоговый код для парсинга ссылок всех объектов в сфере коммерческой недвижимости в Санкт-Петербурге и Ленинградской области.



Данный код включает в себя цикл с искусственными задержками, для прохождения по каждому из 34 объявлений на всех 50 страницах сайта с определенным нами заранее условиями фильтрации. В цикле предусмотрено использование скрипта для вертикального перемещения по странице, с целью загрузки и обработки всех объявления, взятие href-ссылок объектов, то есть url-ссылок опубликованных объявлений, учитывая структуру кода сайта «Циан».

Также было предусмотрено возникновение ошибок, почему после возникновения ошибки сам процесс парсинга не прекращается. Каждая ссылка объявления, при этом, записывается в список и затем сохраняется.

После успешного парсинга ссылок на объекты, возникает надобность парсинга самих объявлений, а для данной работы необходимо провести доскональное изучение кода страницы объявлений чтобы ответить на основные вопросы при такого рода парсинге.

Какие объекты размещены? Какой путь они имеют? В каком формате те или данные записываются? Имеет ли значение в каком порядке эти данные собираются? Обратившись к структуре кода одной из страниц (приложение), была написана программа-парсер (приложение)

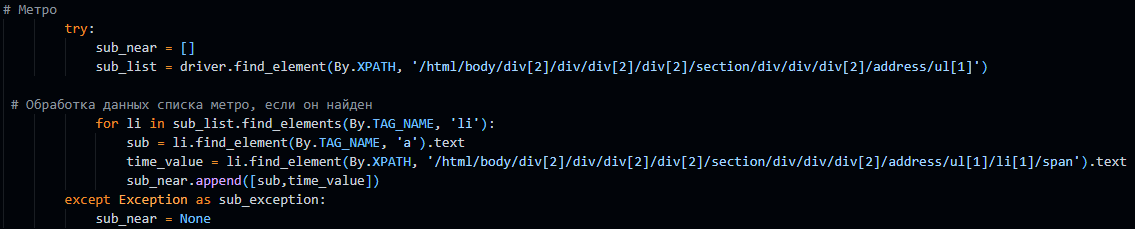
В данном коде происходят действия сбора данных по циклу всех объектов. При открытии каждого объявления, происходит задержка в 10 секунд для полной загрузки страницы и, соответственно, для корректного функционирования парсера. На каждой странице происходит обращение к следующим объектам с извлечением текста и дальнейшим сохранением его, вместе с ссылкой на данное объявление в соответствующем с кодом порядке:

* Название объявления
* Цена объекта
* Цена за квадрат
* Площадь м^2
* Этаж
* Свободно ли помещение
* Город
* Район
* Округ
* Улица
* Номер дома
* Ссылка на страницу продавца
* Описание
* Список ближайших станции метро (мин)
* Список ближайших выездов на шоссе (км).

Работа данного кода суммарно занимает приблизительно 7 часов с учетом обработки всех 1348 объявлений, полученных с работы предыдущего парсера. Такая длительность объясняется наличием задержек для прогрузки страниц и дополнительного времени для сбора информации с каждой страницы.

При изучении написанного парсера, необходимо обратить внимание на структуру внутри цикла. Для каждого из объектов используются операторы “try”, в случае если на какой-либо из страниц не окажется данного объекта или возникнет проблема с его интерпретацией библиотекой selenium. Такая система используется для всех объектов.

Однако ввиду особой структуры объектов, указывающих информацию о расположенных вблизи метро и выездов на шоссе, необходимо было сделать некоторые изменения, а именно:



Поиск элемента, в котором содержатся данные о станциях метро (шоссе), запуск цикла для каждого из элементов в так называемом «списке», затем вывод названия станции метро в отдельную переменную и затем поиск элемента, указывающего на время (км) до метро (шоссе), после чего такой элемент добавляется в список станций метро (выездов на шоссе) у этого объекта.

В случае если у объекта не указаны станции метро (шоссе) – значения столбца остается пустым.

Также стоит отметить, что все эти данные нуждаются в тщательной обработке, ведь все они имеют на данный момент текстовый или даже списочный формат и не подразумевают какой-либо практической ценности.

Рассмотрим пример полученных нами данных:



[]

Обработка первоначальных, так называемых «грязных» данных, является одной из важнейших частей Data-деятельности. Очень важно не потерять данные, обнаружить все ошибки и аномалии, ведь для построения дальнейших моделей нужно иметь подготовленные данные, не вызывающие, как и логических, так и технических конфликтов.

В первую очередь необходимо определить столбцы, которые могут быть полезны для нас и нести смысл, который позволит или более подробно изучить объект или окажется значимым фактором для построения модели.

Одним из важных факторов является площадь самого объекта. Такие данные нам нужны в числовом формате, а именно – float, ведь площадь может быть не целым числом. Для обработки данного столбца используем функции replace и strip и дополнительно библиотеку re для очищения символов единиц измерения площади и любых других символов возле. Используя указанные функции очистим столбец от лишних пробелов и заменим запятые на точки, после чего сможем перевести столбец из str-значения во float. Первоначальная обработка с площадью закончена. Похожие действия проведем и со столбцами полной цены и цены квадратного метра, только вместо единиц измерения площади будут символы рубля.

Обращаясь к следующему столбцу «free» - видим, что свободен ли объект описывается словами – заменяем соответствующие записи на 0 в случае, если объект свободен, и 1, если он занят. При подробном изучении было найдено определенное количество объектов, которые содержали не совсем точные данные касательно занятости объекта, поэтому предварительно были помечены как занятые.

Следующей задачей стояло обработка значений этажа. Согласно сайту «Циан», в объявлении указывался этаж самого объекта, а также максимальный этаж здания, в котором объект находился. Для корректного вывода информации и сохранении этажа и максимального этажа здания – напишем функцию (приложение) для вывода этих данных и создадим два отдельных столбца для них.

Наличие подробного адреса – один из важных фактов выбора именно площадки «Циан», ведь с помощью точного адреса есть возможность преобразовать такого рода данные в координаты и затем использовать их для геоспатического анализа и построения моделей. Для этого необходимо было объединить все столбцы с городом, районом, округом, улицей и домом в одну строку, приведя их при этом в формат, по которому можно было бы найти этот объект на картах.

При попытках обработки столбцов с данными по шоссе возникли проблемы вопроса интерпретации возможного формата обработанных данных. Представленные в виде текста станции метро не позволяли бы учесть их как фиктивные переменные, к тому же был важен фактор дистанции до метро, а циан обозначал его в минутах и мог указывать это время либо как на транспорте, либо пешим ходом, что добавляло проблемы в интерпретации. Такие же проблемы возникли и с шоссе, однако ввиду большего количества уникальных выездов на шоссе – решение такого рода проблем было найдено только для метро.

Учитывая работу с координатами, было решено собрать координаты станций Санкт-Петербургского Метрополитена и затем рассчитать расстояние до объектов недвижимости, а также количество станций метро в радиусе нескольких километров.

[]

Для обработки адресов и перевода их в систему координат было решено обратиться к геокодеру от «Яндекс». Был изучен API геокодера и способы его применения. Первоначально стоит обратиться к термину данной программы.

Геокодер — это программа, которая переводит адрес текстового формата в географические координаты согласно открытой базе данных того или иного источника.

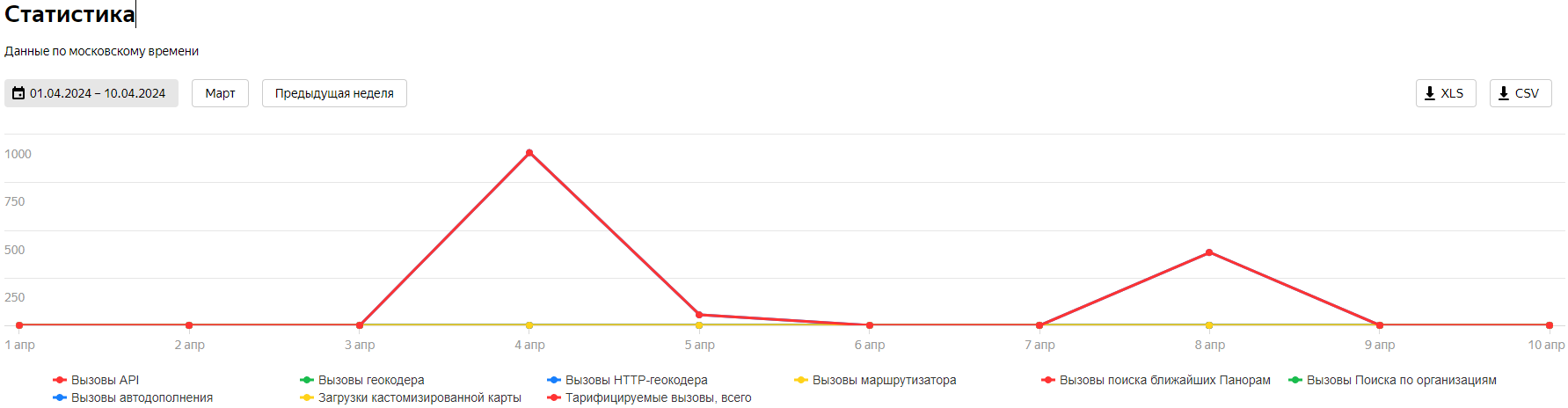
После ознакомления с особенностями геокодера – был создан ключ API для использования его внутри среды кода на языке программирования python и написана функция для отправления запросов в виде адреса и получения результата в формате координат. Также имеются определенные ограничения для бесплатного использования геокодера в рамках научной работы, одним из таких ограничений, оказавших влияние на обработку данных, был лимит на кол-во запросов в сутки – 1000.



Также были учтены возможные возникновения ошибок или факт ненайденных координат.

Данная функция была использована в цикле для получения координат по всему списку данных. Так как данных больше 1000, добавление координат было в 2 этапа. Вместе со вторым этапом также были собраны координаты открытых и ремонтирующихся станций метрополитена в Санкт-Петербурге. На момент отправления запросов – их число составляло 72.

Динамику и статистику запросов также можно рассмотреть в отдельном блоке сайта геокодера.



Для получения максимума пользы из информации по метро, рассчитаем к основным данным расстояние до всех станций метро, чтобы потом мы могли определить, какие станции оставить, а какие убрать. К тому же рассчитаем количество станций метро на определенных расстояниях от объекта. Для этого обратимся к формуле расчета расстояния между двумя точками на координатной плоскости Земли:

Где:

L — искомое расстояние между пунктами 1 и 2;

q — расстояние в одном градусе, константа ~ 111.2 км;

φ₁, φ₂ — географические широты пунктов 1 и 2;

λ₁, λ₂ — географические долготы пунктов 1 и 2.

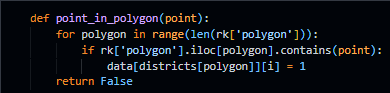
Используя эту формулу, рассчитываем расстояние между каждым объектом и каждой станцией метро, используя циклы. Однако стоит учитывать, что возможно возникновение погрешностей и неточностей.

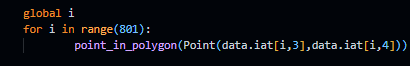
Помимо этого, предварительно была найдена платная база данных по Санкт-Петербургу, включающая основные объекты, по типу зданий, парковок, ж/д путей, дорог и так далее.

Для поставленной задачи – прогнозирования стоимости квадратного метра торгового помещения – наличия парковок является достаточно важным фактором. Это подтверждается и с точки зрения рассуждения, и при обращении к источникам, изучающим факторы влияния на стоимость объектов. Данные по координатам парковок были введены, а затем обработаны с учетом расстояния до объектов. Был добавлен столбец, содержащий количество парковок в радиусе одного километра от каждого из объектов.

К имеющимся данным было решено добавить разделение объектов на районы Санкт-Петербурга, используя фиктивные переменные. За базовое значение было решено взять Центральный район, при этом стоит отметить, что в выборке не оказалось данных по следующим районам: Колпинский, Курортный, Кронштадтский. В связи с чем, данные факторы были убраны из наших данных.

Для выполнения этой задачи был также использован платный набор геопространственных данных по Санкт-Петербургу, разделяющий город на существующие 18 районов. Из имеющегося набора полигонов в рамках системы координат – были собраны названия районов в отдельный список. После чего были импортированы библиотека shapely, позволяющая работать с полигональными и координатными данными. С помощью метода loads данные были конвертированы в формат полигональных данных и затем, была написана функция для проверки, в какой из районов входит объект и последующего обновления столбцов соответствующих районов:

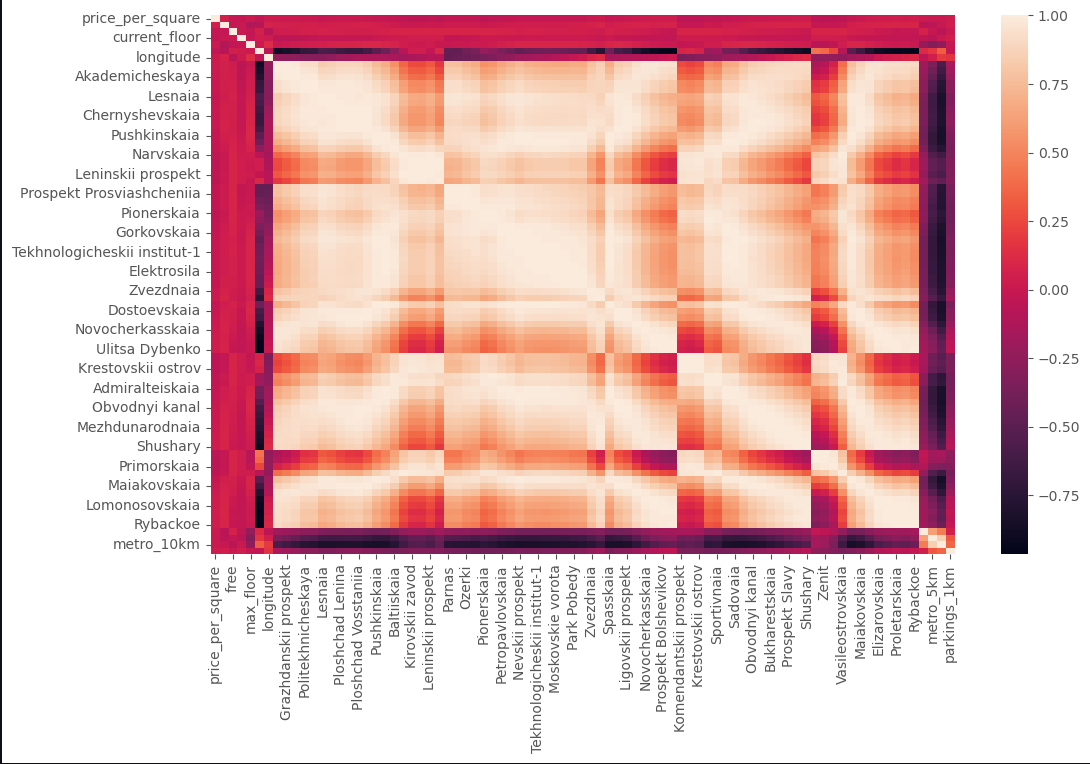




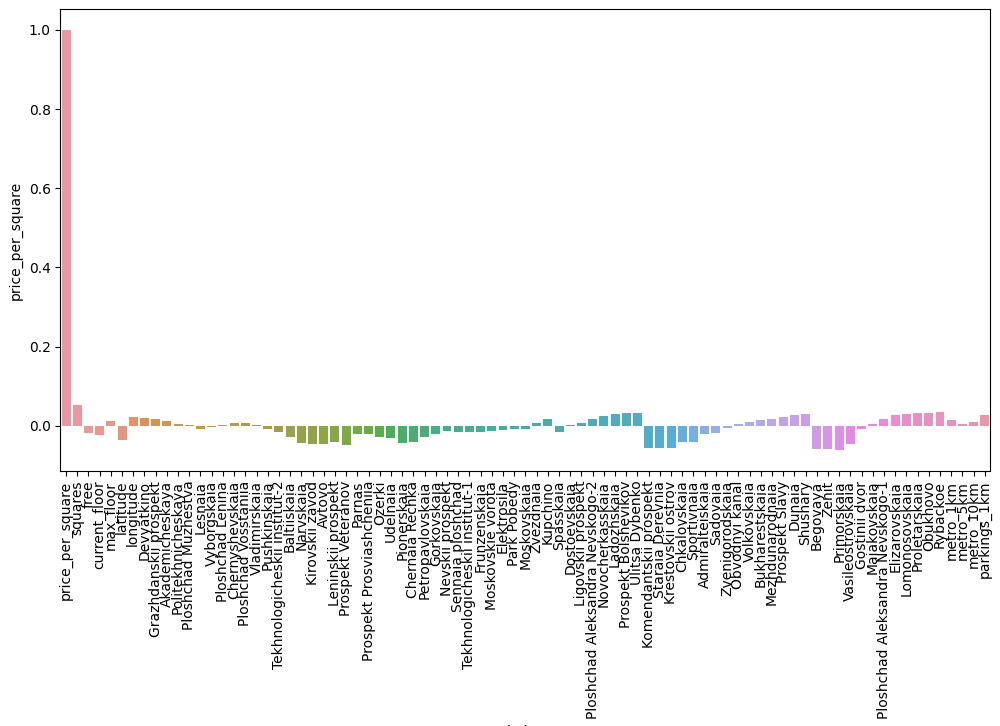
После этого также убираем столбец с центральный районом, для корректного использования фиктивных переменных.

[]

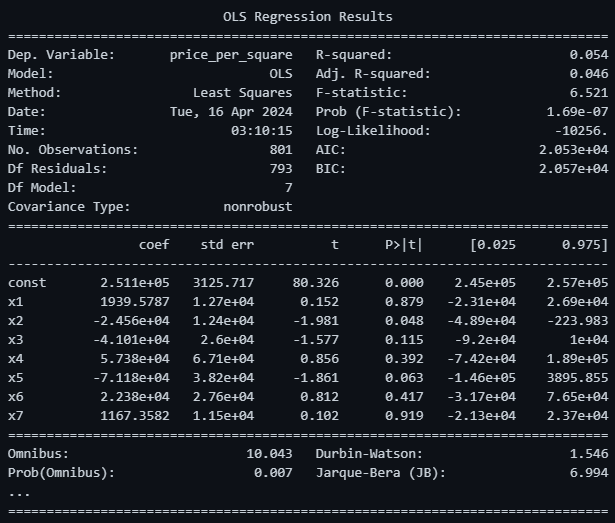
Теперь же возникает немного другого рода проблема – чрезмерное количество факторов в нашем наборе данных, которые при этом создают большую межфакторную корреляцию – дистанции до станций метро. Выведем корреляционную матрицу частей столбцов нашего Dataframe.



Было решено избавиться от большей части признаков на данном этапе модели и в случае необходимости – все же учесть некоторые станции. Также была изучена столбчатая диаграмма, показывающая корреляционную связь с целевым признаком:



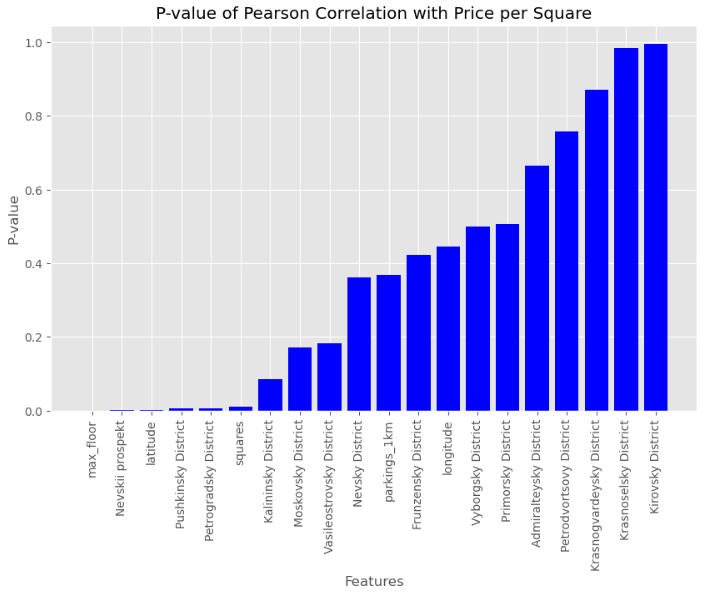
По ней во многом можно сделать заключение о большой роли отрицательного влияния конечных станций на стоимость квадратного метра. Было испробовано несколько вариантов включения определенного количества факторов станций, в особенности использование только конечных станций метрополитена и Невского проспекта, однако во всех вариантах они вызывают корреляцию между собой и не оказывают значимого влияния на все параметры прогнозирования целевого параметра. Это же показывает и вывод регрессионной статистики, созданный для рассмотрения влияния параметров станций метрополитена и демонстрирующий незначимость параметров и неспособность объяснить целевой параметр.



Поэтому было решено на данный момент оставить только станцию метро - Невский проспект, ведь большинство объектов расположены в центре или вокруг центра, а данная станция во многом и представляет собой центр города, что позволит нам также в дальнейшем интерпретировать данный параметр как влияние расстояния от центра Санкт-Петербурга.

Рассмотрим наличие влияния фиктивных параметров между собой и с целевым параметром, а также проверим значимость параметров с учетом также и основных факторов в рамках поставленной задачи.

Для этого мы напишем код, создав отдельный объект – dataframe, в котором с помощью библиотеки scipy используем функцию pearsonr, позволяющую нам вывести p-value для всех используемых факторов:



Анализируя выявленные результаты, можно в целом утверждать, что никакой статистически различимой значимости параметров – фиктивных переменных районов Санкт-Петербурга – не может быть в рамках данной задачи. Возможно, причина лежит в необходимости большего объема равномерных данных – больше помещений в каждом из районов, для обнаружения серьезной зависимости в рамках разных районов.

Учитывая имеющиеся результаты проведенного анализа данных, стоит учесть только самые необходимые параметры, которые показывают наличие статистического смысла в виде зависимости с целевым показателем и наличие логического смысла в рамках данного бизнеса. На данный момент таковыми являются:

* price\_per\_square (Целевой признак, стоимость кв. метра);
* squares (Площадь объекта);
* max\_floor (Максимальный этаж здания);
* longitude (Долгота);
* latitude (Широта);
* Nevskii prospekt (Удаленность от центра);
* parkings\_1km (Количество парковок в радиусе 1 км).

[]

Для дальнейшей работы нам все-таки необходимо снова обратиться к методам обработки данных, однако уже не первичной – необходимо обработать выбросы, аномальные значения и пропуски в столбцах данных, которые могут иметь значимость для нас как исследователей.

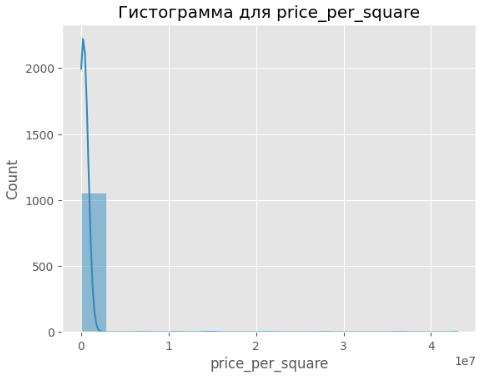
При изучении объявлений и столбцов с данными были обнаружены ошибки в оформлении информации об объектах, в связи с чем цена за квадратный метр была ниже, чем само количество квадратных метров – в связи с чем есть необходимость в удалении указанных данных из нашей выборки. Также, учитывая тенденции рынка и его средние значения, а также консультацию с агентами по недвижимости на предприятии – было решено не учитывать в выборке торговые объекты с квадратурой больше 200 метров квадратных. Они представляют собой несколько иную подкатегорию, нежели изучаемую в данной работе, и имеют очень низкую востребованность ввиду своей дороговизны, в связи с чем гораздо дольше находятся на рынке и корректируют свою цену для привлечения инвесторов или покупателей.

Помимо этого, при определении целевого признака – цены за квадратный метр, возникла надобность в исключении полной стоимости объекта, она бы не позволяла рассматривать построение моделей без учета сильной межфакторной корреляции, по этой же причине данный фактор и не был выбран как целевой.

Целевой признак также нуждается в обработке, ведь помимо обработки аномальных значений, стоит задача максимальной нормализации данных – для улучшения результатов будущих моделей.

[]

При изучении целевого признака, обратим внимание в первую очередь на распределение параметра:



Интерпретация

С учетом такого результата, обратим внимание на аномально высокие и критически низкие показателя стоимости квадратного метра. Стоит при этом учесть, что такого рода данные по большей части были ошибочны определены в исходном источнике, однако даже если некоторые из них не ошибочные – такие данные во всяком исходе не пригодны для нашего исследования, так как попадают кардинально в другое направление коммерческой сферы.

Поэтому мы обрабатываем данные,