Разработка рекомендательного алгоритма для съёмного жилья

Yaroslav Shestakov   
Department of Computer Systems and Technologies  
 National Research Nuclear University MEPhI (Moscow Engineering Physics Institute)  
 Moscow, Russian Federation

[yaroshest@gmail.com](mailto:yaroshest@gmail.com)

Anton Khoroshii

Department of Computer Systems and Technologies  
 National Research Nuclear University MEPhI (Moscow Engineering Physics Institute)  
 Moscow, Russian Federation  
 [horoshiy.ant@gmail.com](mailto:horoshiy.ant@gmail.com)

Nikita Kotlyarov  
 Software for Computers and Automation Systems  
 Bauman Moscow State Technical University (BMSTU)  
 Moscow, Russian Federation  
 [nikita\_kotlyarov2002@mail.ru](mailto:nikita_kotlyarov2002@mail.ru)

***Abstract*—Цель данного исследования - разработка рекомендательного алгоритма для съёмного жилья на основе content-based алгоритма. В больших городах существует проблема с выбором съёмного жилья. Поскольку пользователю сложно определиться с выбором, так как сортировки по стоимости и удаленности от места желаемого проживания не хватает. Каждая квартира по-своему уникальна, от простейших удобств - ванной или стиральной машины - до стиля внутренней отделки. Поэтому необходимо создать систему, которая на основе уже выбранных квартир и их полных характеристик, будет выдавать пользователю наиболее подходящее ему предложение.**

***Keywords— рекомендательный, content-based, съем, квартира***

# Introduction

Данный проект преследует цели облегчить поиск съемного жилья: многие потенциальные потребители не готовы заниматься подобным самостоятельно, ведь такой род деятельности трудоемок – нужно вручную перебрать множество квартир, просмотреть фотографии, описание и комментарии. Мы же готовы предложить программу, способную обучиться вкусам клиента и подобрать оптимальный для него вариант.

В проекте ведется поиск оптимальной рекомендательной системы для поиска жилья.

В исследовании Airbnb-Recommendation [1] прогнозируется оценка пользователя на основе комментариев к обзорам, рекомендует листинг пользователю на основе совместной фильтрации и прогнозирует оптимальную цену. Однако данное решение использует коллаборативную систему – такой подход почти бесполезен в случаях, когда почти нет оценок других пользователей. В случаях с уже готовой базой данных с оценками такой алгоритм наиболее оптимален, но в нашем проекте у каждого пользователя есть только его набор оценок.

Поэтому былорешено использовать content-based рекомендательную систему Пример которой описан статье Bindhu Balu [2]

В сервисе Пик-Аренда [2] представлена похожая по функционалу система, однако из-за небольшого выбора недвижимости и отсутствия информации невозможно определить алгоритм и его эффективность.

# Research Materials And Methods

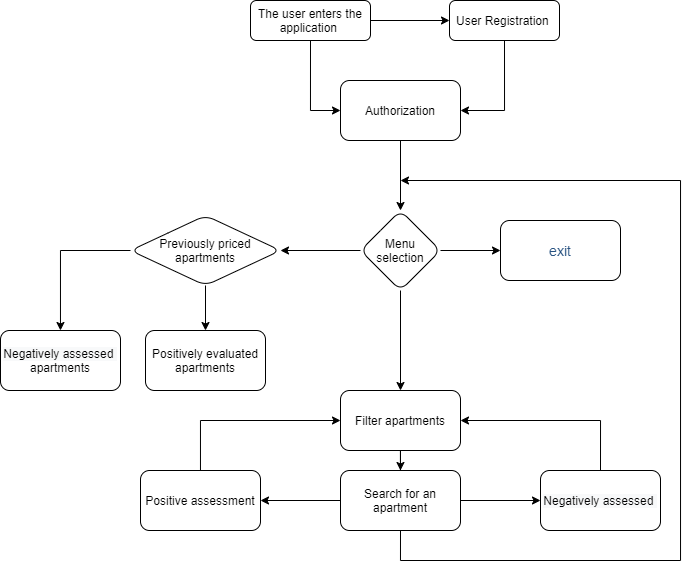
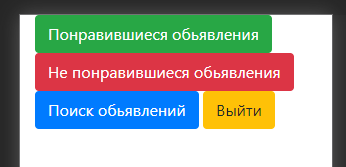
В проекте, прежде всего, используются данные Циана (самая крупная база объявлений об аренде и продаже недвижимости в Московской области). С помощью библиотеки Selenium программа:

1. Переходит на главную страницу Циана и парсит ссылки на объявления сдачи квартир в таблицу данных.
2. Переходит по этим ссылкам и копирует со страниц информацию о соответствующей квартире (описание, предметы быта, цену, фотографии и т.д.).

После сбора информации исследование приступает к этапу обработки: библиотека pymystem3 проводит лемматизацию данных -- избавление от ненужных слов, таких как “и”, “к” и др. и приведение остальных составляющих к нормальной форме (светлые - светлый, ней - она). Это делается ради достижения наивысшей точности при оценке квартиры. После завершения процесса лемматизации всей доступной текстовой информации о квартире, библиотеки numpy и nlrt преобразуют получившийся экстракт в векторное представление. На данном этапе список квартир представляет собой пучок векторов в n-мерном пространстве. Вектор пользователя является средним арифметическим всех векторов квартир, положительно оцененных посетителем. Похожесть имеющихся квартир и “идеальной квартирой” Пользователя-- косинус угла между векторами. Все квартиры сортируются по убыванию косинуса, т.е. от лучшей к худшей и наиболее подходящая выводится на экран.

Алгоритм работы пользователя с сайтом(рис 1): Если пользователь зарегистрирован , он входит в систему. Если пользователь не зарегистрирована в системе , он регистрируется. Далее идет меню(рис 2), где пользователь может выбрать между: просмотром уже оцененных квартир, поиском новой квартиры, и выходом из аккаунта. Для поиска новых квартир пользователю необходимо ввести фильтры для поиска (максимальная цена, станции метро, ремонт). После чего алгоритм предложит оценить квартиру, подобранную по профилю вкуса пользователя. Из меню выбора квартиры пользователь может перейти на страницу объявления в циане и выйти в главное меню.

Сам сайт выполнен на фреймворке Flask это позволило в короткие сроки написать быстроразвертываемый сайт с базой данных и сложной логикой работы, с поддержкой opencv и Python.



Сбор информации, требующейся рекомендательному алгоритму происходил в 2 этапа:

1. Копирование ссылок на объявления.
2. Переход по ссылкам и дальнейшее скачивание информации о квартире.

Благодаря ресурсоемкому устройству сайта Циана, отображаются не все квартиры, а только выборка из различных ценовых категорий, помещающаяся на 54 страницах - по 28 квартир на каждой. Таким образом становится невозможным банальное перемещение по страницам сайта и копирование ссылок. В результате приходится вводить ценовые рамки квартир, чтобы их количество не превысило 1512. Проект вводит в поля “Цена от - до” числа, начиная с 0 - 27000 и идя с интервалом по 1000 руб.: 27001 - 28000 и тд.

Также возникли проблемы с перелистыванием страниц на сайте - у Циана есть всплывающие окна, препятствующие нажатию нужных кнопок.



Функция selenium “move\_to\_element” -- самая используемая функция перемещения мышки к элементу на экране -- не справлялась с поставленной задачей. С трудом был найден процесс прокрутки экрана, привязанный к определенному объекту на странице. В остальном проблем со скачиванием ссылок на квартиры не возникло.

Сбор данных отдельных квартир происходил по по другой схеме. Программа переходит по ранее спарсенным ссылкам на объявления о сдачи квартиры на съем и скачивает всю нужную информацию: описание, адрес, имеющиеся теги (стиральная машина и тд.) и, естественно, фотографии самой квартиры.

Также циан использует сложную систему хранения фотографий , вследствие чего невозможен быстрый парсинг объявлений. Вот пример путей по которым хранятся фотографии в памяти агрегатора недвижимости.





Фотографии хранящиеся в разных папках с примерно одинаковыми , но различными путями. В рамках данного проекта не удалось выявить зависимость адреса фото от адреса объявления к которому принадлежат искомые фотографии.

Все данные системы хранятся в базе данных SQL, для комфортного взаимодействия пользователя с системой. Взаимодействие с базой данных происходит посредством ORM базы данных sqlalchemy. Каждая операция с базой данных производиться в отдельной сессии в объекте базы данных. Все векторы заранее нормализуются и заносятся в базу данных в виде бинарного файла. Из-за этого база данных на 3000 квартир весит около 200 мб за счет этого возможна быстрая выдача квартиры пользователю при использовании сайта. Был добавлен фильтр по ремонту, исходя из статьи Peter Bondarenko [3], была обработана вся база данных с помощью нейронной сети описанной в статье. И добавлена дополнительная функция поиска по ремонту. При полной базе данных фотографий квартир , появиться возможность увеличить точность предсказаний нейронной сети.

Алгоритм выдачи рекомендации(рис 1) следующий:

1. От пользователя приходят фильтры (максимальная цена, станции метро, ремонт).
2. По фильтрам пользователя отбираются апартаменты , которые пользователь еще не оценил.
3. Если их нет , то пользователю предлагается сменить фильтры или просмотреть список ранее оцененных квартир.
4. Дальше программа собирает список квартир , которые пользователь уже оценил, чтобы на их основе выдал новую квартиру, если пользователь еще ни разу не оценивал, квартир ему предоставляеться случайная, удовлетворяющая фильтрам.
5. Далее создается нормализованный вектор оцененных квартир пользователя. Вектор пользователя сравнивается с каждым вектором квартиры, подходящей по фильтрам. Самый близкий вектор к вектору пользователя возвращается сайту и он предоставляет возможность пользователю оценить новую квартиру.

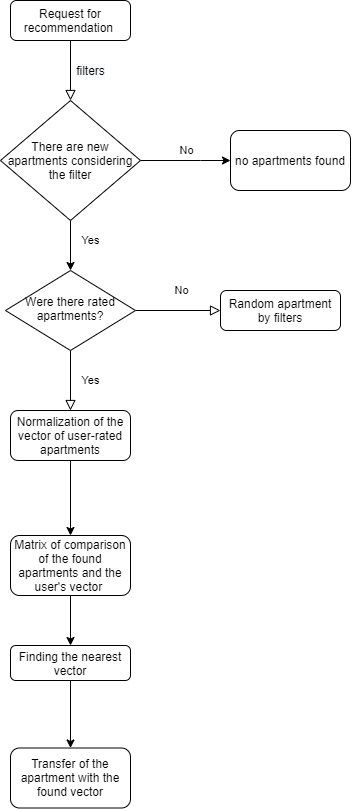


Рис 1. Алгоритм выдачи рекомендации

При обработке базы данных, возникли проблемы с скоростью обработки каждой квартиры. Лемматизация длилась до 10 секунд. Так же и обработка могла занимать до 30 секунд. Чтобы ускорить процесс обновления базы данных, были написаны программы по распараллеливанию процессов с помощью библиотеки Joblib [4], лемматизации и классификации изображений в параллельных потоках. Что сократило время обновления базы данных от предполагаемых 40 часов до 20 минут на имеющемся персональном компьютере.

Также в целях оптимизации изображения квартир не храняться в проекте, в базе данных квартир в виде списка, хранятся лишь ссылки на изображения со страниц агрегатора недвижимости. Это позволяет сэкономить на хостинге и ускорить работу пользователя с приложением.

# Results

Благодаря оптимизации парсинга удалось сократить время парсинга c 4 дней до 2 а именно: все страницы на которых возникли ошибки парсинга помечаются как битые и больше не попадаются при перезапуске парсинга. Так же проверяеться наличее данных о квартире в базе данных, если они уже внесены, то еще раз парситься квартира не будет. Примеры данных в базе данных: ссылки на объявления(Рис.7), данные объявлений (Рис. 8)

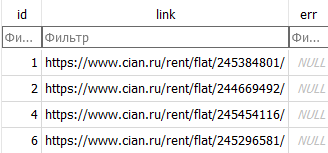


Рис.7 Ссылки на объявления

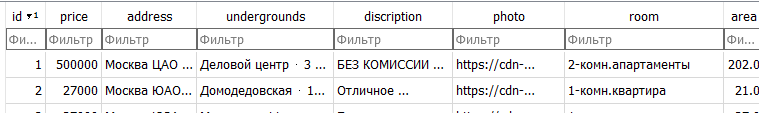


Рис.8 Данные объявлений

Пример рекомендации выданный приложением (Рис. 9)

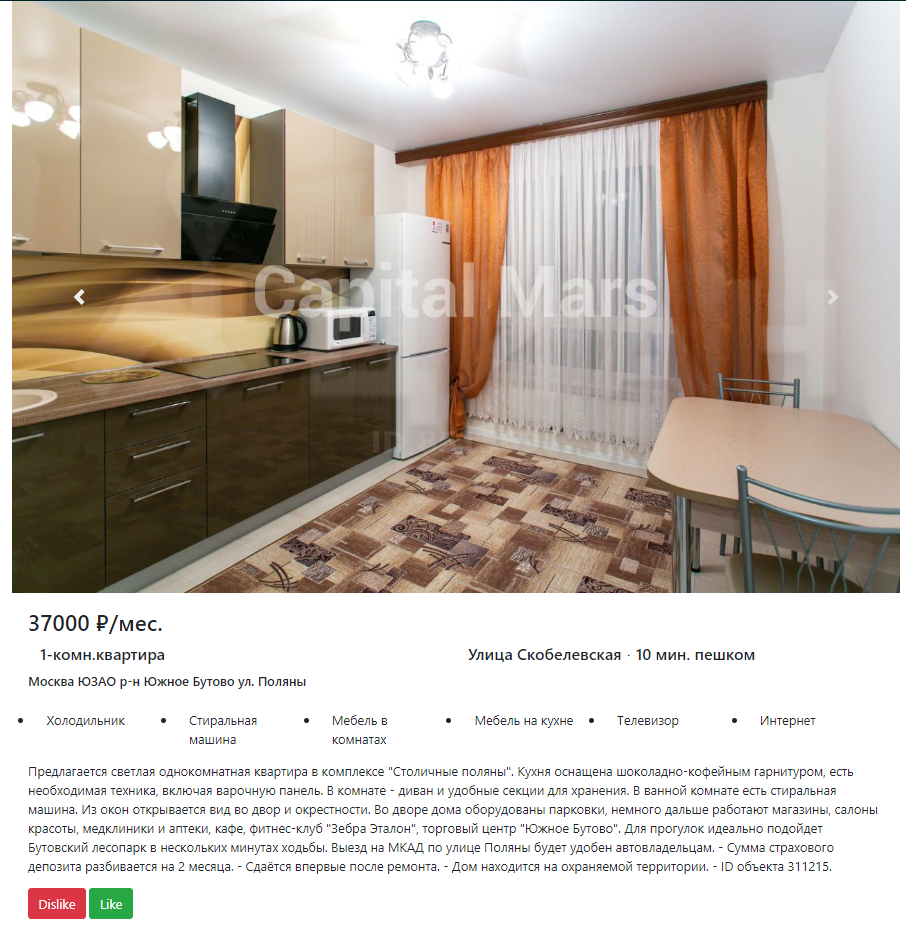


Рис. 9 Пример рекомендации

Также пользователь может посмотреть уже оцененные квартиры (Рис. 10): Как выглядят оценки в базе данных (Рис.11)

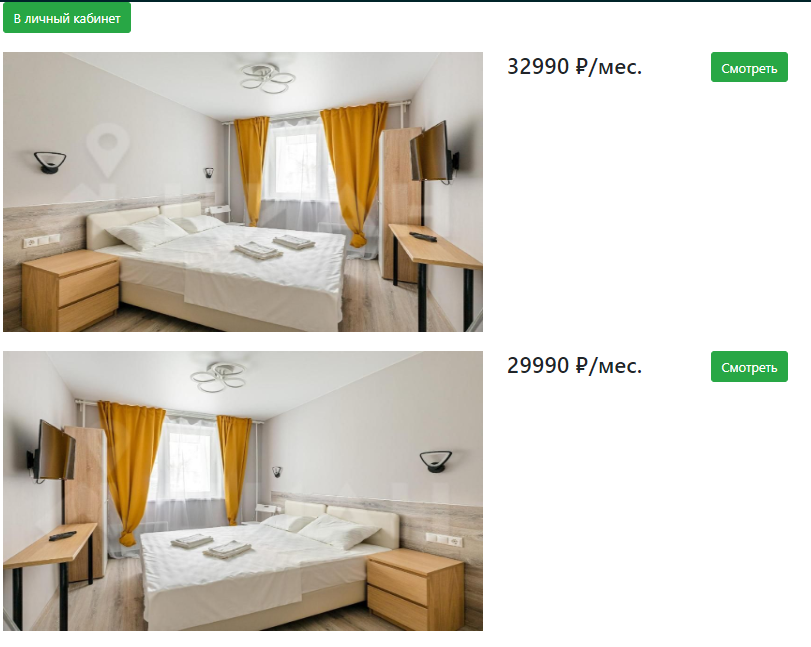


Рис. 10 оцененные квартиры на сайте

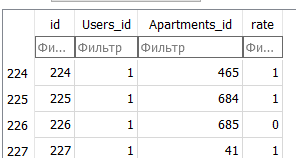


Рис. 11 Оцененные квартиры в базе данных

# Discussion And Conclusions

В результате разработки был создан алгоритм, позволяющий внедрить рекомендательную систему по поиску съемной квартиры в уже существующие агрегаторы. Благодаря данному проекту стал возможным искусственный подбор нужному пользователю жилищу. Также необходимо отметить гибкость алгоритма: при желании и минимальных усовершенствованиях становится доступным рекомендательный алгоритм не только для квартир, но и практически для любых видов товаров и даже услуг. При обмене опытом с экспертами в требуемых областях станет возможным создание дополнительного модуля проекта -- рекомендации определенного рода услуг. Таким образом потенциал рекомендательной системы безграничен. В проекте не используются уже известные величины привлекательности (оценки и отзывы других пользователей) и большой базы данных зарегистрированных пользователей.

Проект призван облегчить поиск съемного жилья в Москве пользователям. Было создано веб приложение для поиска съемной квартиры при помощи рекомендательного алгоритма. Адаптировано для скачивания на телефоны. Реализованная регистрация и хранение данных квартир и пользователей.

##### References

1. DeveloperManisha. Airbnb-Recommendation-System. 2018, Available at: https://github.com/DeveloperManisha/Airbnb-Recommendation-System (Accessed 28 November 2020).
2. Bindhu Balu, Content-Based Recommendation System, 2019, Available at: https://medium.com/@bindhubalu/content-based-recommender-system-4db1b3de03e7 (Accessed 28 November 2020).
3. *PIK-Arenda,2020,* Available at: https://play.google.com/store/apps/details?id=ru.pik.arenda.selection&hl=ru(Accessed 28 November 2020). (In Russian).
4. Peter Bondarenko. Kak ja nashjol krasivuju kvartiru s horoshim remontom s pomoshh'ju sobstvennoj nejroseti. 2020 . Available at: https://vc.ru/ml/138697-kak-ya-nashel-krasivuyu-kvartiru-s-horoshim-remontom-s-pomoshchyu-sobstvennoy-neyroset (Accessed 28 November 2020). (In Russian).
5. Joblib, Available at: https://joblib.readthedocs.io/en/latest/ (Accessed 28 November 2020).