**Федеральное государственное автономное образовательное учреждение высшего образования**

**«Национальный исследовательский университет**

**«Высшая школа экономики»**

**Факультет компьютерных наук**

**Основная образовательная программа Прикладная математика и информатика**

**КУРСОВАЯ РАБОТА**

**Программный проект на тему**

**«Компьютерный ассистент менеджера по подбору площадок для киносъемок»**

Выполнила:

Студентка 3 курса БПМИ 188 Квиндт Ева Сергеевна

Руководитель КР:

Младший научный сотрудник МЛ ИССА Паринов Андрей Андреевич

**Москва 2021**

**Оглавление**

[Аннотация 4](#_Toc73824966)

[Ключевые слова 4](#_Toc73824967)

[Annotation 4](#_Toc73824968)

[Keywords 4](#_Toc73824969)

[Введение 5](#_Toc73824970)

[Существующие подходы 6](#_Toc73824971)

[Выводы 7](#_Toc73824972)

[Векторное представление текстовых описаний 8](#_Toc73824973)

[Doc2Vec 8](#_Toc73824974)

[Снижение размерности пространства 10](#_Toc73824975)

[Principal component analysis 10](#_Toc73824976)

[T-distributed Stochastic Neighbor Embedding 11](#_Toc73824977)

[Uniform Manifold Approximation and Projection 11](#_Toc73824978)

[Linear Discriminant Analysis 11](#_Toc73824979)

[Singular Value Decomposition 12](#_Toc73824980)

[Вывод 12](#_Toc73824981)

[Сбор и анализ данных 13](#_Toc73824982)

[Парсинг данных 13](#_Toc73824983)

[Анализ и обработка данных 14](#_Toc73824984)

[Кластеризация объектов 16](#_Toc73824985)

[K-Means 16](#_Toc73824986)

[Hierarchical clustering 16](#_Toc73824987)

[DBSCAN 16](#_Toc73824988)

[Кластеризация данных 17](#_Toc73824989)

[Рекомендательная система 19](#_Toc73824990)

[Заключение 20](#_Toc73824991)

[Дальнейшая работа 20](#_Toc73824992)

[Список источников 22](#_Toc73824993)

# Аннотация

На данный момент индустрия кино нуждается в создании уникальной платформы, где менеджеры по подбору локаций могут выбирать места съемок для своего проекта.

В данной работе будет упомянуто о причинах необходимости такого приложения, и в частности речь пойдет о построении рекомендательной системы для онлайн-помощника, который параллельно создавался другой командой разработчиков.

В работе приведено описание аналогов рекомендательных систем, проблем, методов решения, а также пошаговое построение рекомендательной системы, в том числе анализ предоставленных данных.

# Ключевые слова

Рекомендательная система, фильтрация на основе содержания, снижение размерности, анализ и визуализация данных, векторное представление слов, embedding, кластеризация.

# Annotation

At the moment, the film industry needs to create a unique platform where film managers can choose filming locations for their project.

In this paper, we will mention the reasons for the need for such an application, and in particular we will talk about building a recommendation system for an online assistant, which was created in parallel by another development team.

The paper describes the analogues of recommendation systems, problems, methods of solution, as well as the step-by-step construction of the recommendation system, including the analysis of the data provided.

# Keywords

Recommendation system, content based filtering, dimensionality reduction, data analysis and visualization, embedding, clustering.

# Введение

В последнее время кинематограф стал одним из главных направлений в развитии сферы развлечений. Ежегодно выпускаются сотни новых фильмов и сериалов, а согласно мнению экспертов, 2021 год может стать рекордным для российской киноиндустрии. Поход в кинотеатр – один из самых бюджетных способов массового развлечения на данный момент, поэтому на фоне снизившейся после пандемии покупательской способности, он будет и самым востребованным. Отметим, что российские кинофильмы являются неотъемлемой частью данной сферы. Это можно видеть по данным Фонда кино: за 2020-ый год доля российских фильмов составила около половины от всей массы премьер на российском рынке.

Несмотря на то, что подбор и утверждение кинолокаций – одна из главных компонент в процессе создания кино, на сегодняшний день в российской киноиндустрии нет одного универсального инструмента, позволяющего это сделать. Сейчас процесс имеет следующий вид:

1. Осуществляется поиск площадок на различных источниках. Например, babooshka.pro, locationhunters.ru, kinolocation.ru, kino-bank.com, kinopartner.ru и другие.
2. Составляется и отдается на утверждение продюсеру шорт-лист из наиболее подходящих вариантов со всех источников.
3. Продюсеры выбирают из этого списка необходимые площадки и передают эту информацию обратно менеджерам.
4. Менеджеры связываются с арендодателями объектов, используя разные источники размещения информации об этих площадках, и договариваются об их аренде.

Большинство из вышеописанных сервисов с информацией о площадках не позволяют менеджерам авторизовываться в системе, просматривать объекты на карте, использовать мобильное приложение, получать рекомендации от системы и др. Однако, именно эти возможности являются ключевыми для менеджеров и продюсеров.

В отличии от конкурентов программная система подбора площадок для киносъемок будет иметь весь вышеперечисленный функционал. В частности, в нее будет встроена рекомендательная система, которая упростит процесс подбора площадки за счет того, что будет предлагать пользователю потенциально подходящие ему варианты, основываясь на характеристиках площадок и предпочтениях пользователя.

# Существующие подходы

Задача построения рекомендательной системы сильно зависит от объекта, который система предлагает пользователю: маршрут поездки, фильм к просмотру, квартира и др. От объекта рекомендации зависит структура данных, с которыми предстоит работать, информация о пользователе, которую нужно собрать и, собственно, устройство самой системы.

Существуют два подхода построения рекомендательных систем. В первом, специалисты сервиса сами указывают, какие категории товара могут предлагаться параллельно с выбранным, основываясь на категориальных признаках объектов. Во втором случае, используется анализ данных и машинное обучение. Строятся точные или самообучающиеся модели рекомендаций, основанные на различных данных об объектах и пользователях. Конечно, мы будем рассматривать последний подход исходя из двух его преимуществ. Во-первых, он позволяет улавливать неочевидные взаимосвязи между объектом и пользователем. Во-вторых, в нашем сервисе не предусмотрена возможность ручной разметки объектов на похожие и различающиеся. К тому же, не ясно, как определять «похожесть», учитывая, что на рассмотренных сервисах почти отсутствуют категориальные характеристики площадок.

Две основные стратегии создания рекомендательных систем, использующих точные математические модели, это коллаборативная фильтрация и фильтрация на основе содержания. В первом методе используется информация о прошлых действиях пользователей (например, какие оценки они ставили продуктам). Производится сбор оценок объектов пользователями и на их основе строится строится рейтинг объектов или вычисляется вероятность положительной оценки от конкретного пользователя для данного объекта. Это универсальный и наиболее популярный подход, однако у него есть большой минус – «холодный старт». Когда появляется новый пользователь или объект, у нас отсутствует какая-либо информация, необходимая для работы модели.

При фильтрации на основе содержания (content based filtering) создаются профили объектов, содержащие такие характеристики, как жанр, текстовое описание, демографическая информация, имена актеров и т.д. Пользователю предлагаются варианты, сгруппированные по характеристикам, или похожие на те, которые он просматривал ранее. Например, рекомендация фильмов в жанре Хоррор или классические брюки других брендов. В этом случае большое имеет значение структура объекта и предметная область, с которой мы работаем. Минус такого подхода заключается в том, что моделью могут не учитываться скрытые зависимости между предпочтениями пользователя и объектами, в отличие от вычислительных методов коллаборативной фильтрации.

При построении любой рекомендательной системы, конечно, нужны данные для тестирования и понимания предметной области. При их сборе обычно используются сочетания явных и неявных методов. Явные методы основаны на прямых вопросах пользователю, касательно его предпочтений. Сюда же можно отнести опросники на сайтах с просьбой поставить оценку просмотренному контенту или купленному товару. Такой подход дает возможность четко оценить отношения пользователя к объетку.

При неявном сборе данных мы анализируем поведение клиентов на сервисе или, например, содержимое их устройств. Однако, это не самая точная информация о предпочтениях. Например, если пользователь просмотрел фильм, то это означает лишь то, что фильм казался ему достаточно интересным на этапе осуществления выбора, но, возможно, не понравился после.

## Выводы

Логика, на которой будет основываться рекомендательная система ограничивается только фантазией разработчиков. Достаточно взглянуть на варианты различных рекомендаций известных музыкальных сервисов: персональная подборка того, что вам нравится; топ популярных песен за 2021 год; треки, которые вы слушали в 14 лет и которые сформировали вас; подборки по настроению.

В нашем кейсе мы строим рекомендательную систему площадок для киносъемок. Как говорилось ранее, универсального сервиса по подбору кинолокаций в России нет, а существующие предоставляют только краткое текстовое описание площадки, фотографии и, если повезет, информацию о местоположении. Рекомендательные системы на подобных сервисах не представлены, поэтому опираться можно лишь на смежную область (например, рекомендации квартир на сервисах по аренде недвижимости). К тому же у нас нет возможности собрать информацию о пользователях, так как сервис находится в разработке.

Основываясь на всем вышесказанном, мы поставили задачу создать рекомендательную систему, которая бы по истории посещений пользователя предлагала похожие варианты на основе текстового описания площадки. Выбор текстового описания в качестве характеристики, на которой основывается рекомендательная система сделан не случайно. Из всех имеющихся данных об объектах, текстовое описание или название – ключевые и практически единственные атрибуты, которые присутствуют на всех сервисах. Логика работы рекомендательной системы основывается на методе фильтрации на основе содержания.

# Векторное представление текстовых описаний

Для построения рекомендательной системы на основе описания площадок для киносъемок необходимо привести текстовое описание к понятному для машины представлению – численному. В этом пригодятся методы векторного представления слов (word embeddings).

В процессе работы были рассмотрены лишь некоторые из них: One Hot Encoding (OHE), Bag of Words (BoW), Word2Vec, Doc2Vec, FastText, Term Frequency — Inverse Document Frequency (TFIDF). Важно было подобрать метод, который можно реализовать на Python, но главное, чтобы он подходил для поставленной задачи и имеющихся данных.

Описание площадок чаще всего представлено двумя-тремя предложениями с главной информацией об объекте: тип (лофт, крыша, парковка), местоположение (Москва, Московская область, ЦАО), окружающая среда, наполнение (большие окна, оборудование для освещения). Поэтому методы, работающие с большими корпусами, такие как FastText и TFIDF, не подходят для использования. В частности, FastText содержит в себе предобученные готовые векторные представления слов и классификатор. Однако, в данном случае эти преимущества становятся недостатками, так как ожидается, что заранее предобученные данные плохо справятся с грамматическими и синтаксическими ошибками.

OHE и BoW наоборот слишком примитивны для данной задачи. Они просты в реализации, но не эффективны по памяти. А главный минус – не учитывают контекст и порядок слов, только наполнение предложения. Для нашей задачи это существенный недостаток, так как предложения состоящие из похожих наборов слов могут описывать различные помещения. (Например, «большая площадка на открытом воздухе для съемки кино» и «съемная квартира с большими окнами и воздушным интерьером»).

Наиболее подходящими методами стали Word2Vec и Doc2Vec, но предпочтение было отдано последнему.

## Doc2Vec

Каждой площадке в нашей задаче сопоставляется ее текстовое описание в виде нескольких предложений. Для построения рекомендательной системы необходимо понимать, какие площадки похожи друг на друга, то есть определять близость описаний друг к другу.

В отличие от Word2Vec, цель Doc2Vec – получать числовое представление документа, а не только отдельных слов. В своей структуре эти два метода похожи, однако Doc2Vec добавляет вектор признаков, уникальный для каждого документа.

# Снижение размерности пространства

Снижение размерности пространства – это преобразование данных путем уменьшения числа переменных за счет выделения или создания главных компонент, содержащих ключевую информацию об объектах. Например, если характеристика представлена недостаточно полно (много пропусков, неоднородность ее показателей) или не несет за собой данных об объекте (например, зачастую – уникальный ID клиента) – релевантно использование алгоритма снижения размерности.

Данные методы применяются для уменьшения времени обучения моделей и расходуемой памяти, повышения качества результатов модели, сжатия изображений и для упрощенного представления данных – визуализации (сведение размерности к минимальной, такой как 2D или 3D).

Алгоритмы понижения размерности в основном сохраняют либо глобальную структуру данных, либо локальные расстояния между точками. К первым относится Метод главных компонент (PCA) и MDS (Multidimensional Scaling), а ко вторым — t-SNE, ISOMAP, LargeVis, UMAP и другие.

В нашей работе методы снижения размерности понадобились для визуализации данных. Необходимо было понимать, с какой структурой приходится работать и в дальнейшем подобрать подходящий метод кластеризации.

## Principal component analysis

Один из основных и самых простых методов линейного уменьшения размерности данных. Зная зависимости между компонентами и их силу, выражает несколько признаков через один. Максимизирует дисперсию и старается сохранить большие попарные расстояния между объектами.

Для начала PCA стандартизирует и нормирует все признаки. Затем вычисляет ковариационную матрицу, которая описывает разброс случайной величины (новой характеристики). Сортирует собственные вектора по значениям их собственных значений в убывающем порядке и выбирает первые k штук, которые и составляют полученное новое пространство признаков.

За работой данного метода стоит идея о том, что между компонентами есть скрытые корреляции, которые необходимо выявить. Компоненты, с сильными корреляциями можно выразить друг через друга и оставить тем самым только линейно независимые признаки, полностью описывающие изначальную структуру данных.

Данный метод плох для визуализации данных, так как при редукции до двух компонент, отбрасывает локальные различия между объектами. Именно поэтому чаще всего PCA используется для снижения размерности в целях повышения эффективности работы дальнейших моделей (ускорение по времени и уменьшение используемой памяти). Зачастую PCA применяют к датасетам перед другими, более «тяжелыми», методами снижения размерности.

## T-distributed Stochastic Neighbor Embedding

Данный метод, в отличие от PCA, сохраняет попарные сходства между объектами. Сходство точек вычисляется как условная вероятность того, что точка А выберет другую точку в качестве своего соседа. Алгоритм сводит к минимуму разницу между этими условными вероятностями в пространстве более высоких и более низких измерений для идеального представления точек данных в пространстве более низких измерений.

Очень часто t-SNE используется поверх PCA. Хорошо подходит для визуализации данных большой размерности и чаще всего для этого и используется.

## Uniform Manifold Approximation and Projection

Сравнительно новый метод уменьшения размерности, напоминающий по своей структуре t-SNE. Как заявляют его авторы, он превосходит предыдущие методы (в частности t-SNE) в качестве и скорости работы и эффективней переносит глобальную структуру данных в новое уменьшенное пространство признаков.

UMAP рассчитывает расстояние между объектами по заданной метрике и находит n ближайших объектов для каждого. Далее алгоритм строит взвешенный граф, соединяя ребрами этих соседей. Затем создает граф в пространстве меньшей размерности и приближает его к исходному, минимизируя сумму расстояний Кульбака-Лейбера.

## Linear Discriminant Analysis

LDA использует информацию от нескольких признаков для создания новой оси и проецирует данные на эту новую ось таким образом, чтобы минимизировать дисперсию и максимизировать расстояние между средними двух классов. Цель LDA – сохранить компоненты, в которых находится информация о различиях между разными кластерами в данных.

Для начала алгоритм рассчитывает среднее значение каждой переменной по каждому классу. Затем вычисляет матрицу рассеивания внутри класса и между классами, вычисляет их собственные значения и вектора, сортирует в порядке убывания и выбирает первые k штук. Создает новую матрицу, содержащую собственные векторы, которые сопоставляются с k выбранными значениями и получает новые признаки, беря точечное произведение данных матриц.

Метод показывает эффективные результаты в случае равномерно распределенных данных, иначе имеет тенденцию к смещению в сторону класса большинства. Поэтому для LDA данные нужно тщательно подготовить, что уже является огромным минусом. Но главный недостаток заключается в том, что алгоритм работает только на размеченных датасетах.

## Singular Value Decomposition

Из линейной алгебры известно, что любую матрицу можно представить в виде произведения трех: унитарной матрицы U порядка m, матрицы Σ размера m x n, на главной диагонали которой лежат неотрицательные числа, называющиеся сингулярными и матрицы V\* (сопряжённая к V матрица порядка n).

Матрица Σ из полученного разложения пригождается в задаче редукции. Ее элементы, возведенные во вторую степень, интерпретируются как дисперсия, которую «вкладывает» каждая характеристика (компонента). Причем элементы расположены в убывающем порядке, что позволяет просто выбрать первые k штук.

SVD зачастую используется для работы с изображениями (сжатием и восстановлением картинки), эффективен по времени, однако для визуализации плох, так как первые две-три компоненты не содержат в себе много дисперсии от изначальных данных.

## Вывод

Выбор метода снижения размерности зависит от данных, с которыми предстоит работать. Заранее можно было сделать лишь предположения о том, какие методы могут подойти для получения качественной визуализации, в которой как можно точнее сохранялись бы и внутриклассовые и межклассовые различия.

Универсальным методом оказался TSNE и UMAP. Они эффективны по времени, имеют простую реализацию на Python, не требуют размеченных данных (в отличие, например, от LDA), хорошо подготавливают данные для дальнейшей кластеризации и сохраняют много информации от изначальных данных.

# Сбор и анализ данных

## Парсинг данных

Парсинг данных осуществляли коллеги по разработке сервиса по подбору площадок. Однако стоит уточнить, откуда именно и как появились данные для построения рекомендательной системы.

Было произведено сравнение объявлений о площадках для съемок со следующих веб-страниц: https://babooshka.pro/, https://locationhunters.ru/, http://kinolocation.ru/, https://kino-bank.com/, <https://kinopartner.ru/>. На основе этого выявлены атрибуты объекта «Локация»:

• Id – идентификатор объекта;

• Name – название объекта;

• Description – описание объекта;

• Address – адрес объекта (точного адреса нет ни на одном источнике);

• Source – название сайта-источника локации;

• Images – изображения локации;

• Imported – флаг о том, что локация является импортированной.

Отметим, что из всех атрибутов обязательным для всех объектов являлся только один - Name. При обучении данный атрибут использовался наравне с Description как описание объекта.

В модель данных также были добавлены дополнительные атрибуты, которых не было у объявлений с рассмотренных источников, однако в теории такие атрибуты могут быть на других сайтах-источниках:

• OwnerPhone – телефон владельца локации;

• OwnerName – имя владельца локации;

• CoordLat – географическая широта локации;

• CoordLon – географическая долгота локации;

• Tags – тэги, по которым можно найти объект;

• Parent – родительский объект «Локация»;

• Manager – пользователь, работающий с объектом.

В качестве фреймворка для веб-скрапинга был выбран Scrapy, в качестве паттернов парсинга данных – XPath и Regexp. Для сериализации полученных данных в формат JSON использовалась библиотека Python json.

## Анализ и обработка данных

После парсинга данных был получен json файл с описаниями площадок в виде {id, description}. Для дальнейшей работы с текстом, все буквы переведены в нижний регистр, удалены знаки препинания, табуляции и переноса строки. Также описания очищены от абревиатур округов и районов, так как описание объекта, на котором строится рекомендация, не подразумевает под собой информацию о местоположении. При наличие данной информации, есть риск испортить модель, так как она будет полагаться не на схожесть объектов по виду и наполнению, а на месторасположение.

Также удалены пустые и тестовые строки.

В итоге получено 698 строк, 442 из которых – уникальные.

Полученные данные переведем в векторное представление при помощи ранее выбранного Doc2Vec.

Настройка параметров производилась опытным путем и на основе рекомендаций по работе с Doc2Vec с электронных ресурсов. Параметр *vector\_size,* векторного представления описания не должен сильно превосходить размер текстов. Обратное может привести к появлению переменных, не несущих в себе исчерпывающей информации об объекте. *window* равен трем из подобных соображений – мы не хотим, чтобы модель считывала контекст по всему предложению. *dbow\_words* установленный в значении *False* позволяет оптимизировать модель, не сохраняя веса отдельных слов, а обучая только векторы предложений.

После обучения Doc2Vec можно посмотреть на то, какие предложения модель определила как близкорасположенные (см. Рисунок 1-3).

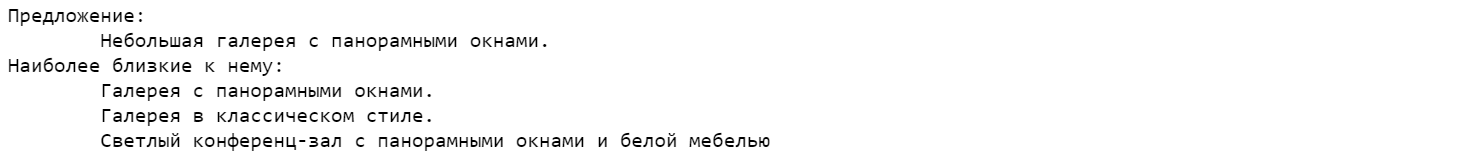


Рисунок 1

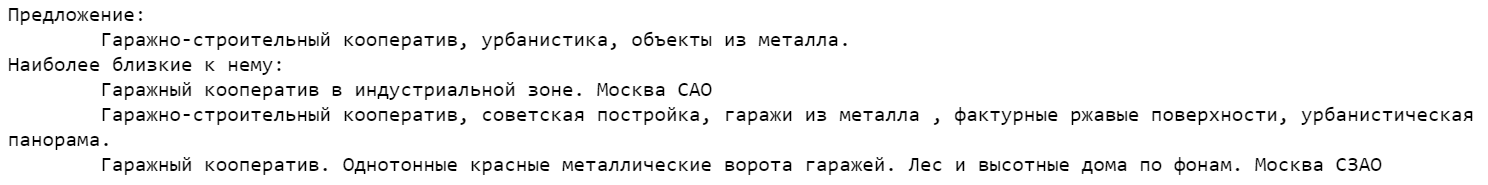


Рисунок 2

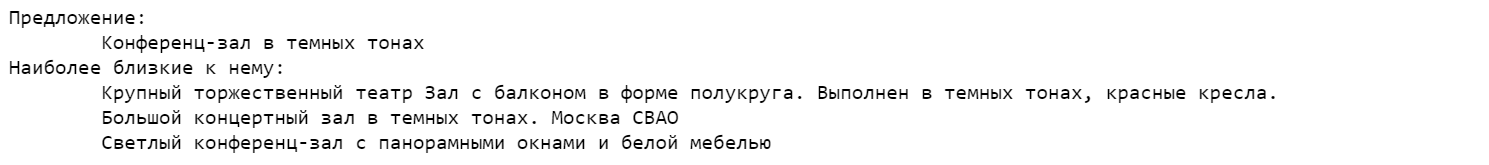


Рисунок 3

Далее произведено снижение размерности пространства двумя выбранными способами: при помощи UMAP и при помощи TSNE (использовать PCA перед этим не было необходимости, так как датасет небольшой и векторы описаний тоже небольшой размерности (20). Результаты визуализации представлены на Рисунке 4.

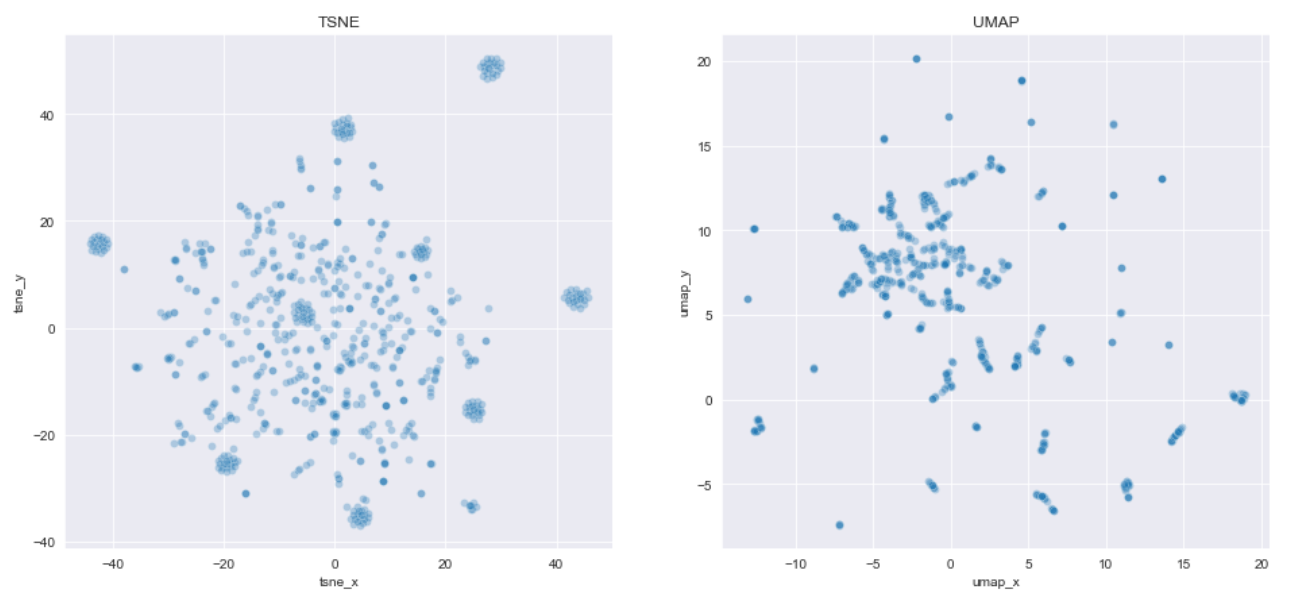


Рисунок 4. Снижение размерности пространства двумя способами (слева - TSNE, справа - UMAP)

В целом, результаты обоих методов получились достаточно хорошими для дальнейшей кластеризации. Однако, TSNE лучше сгруппировал описания. Объекты в данном случае расположены дальше друг от друга (видно по шкалам осей), кластеры больших мощностей, кластеры имеют круглую форму, что облегчит кластеризацию.

# Кластеризация объектов

Для задачи кластеризации были рассмотрены несколько алгоритмов: K-Means, Ward, DBSCAN.

## K-Means

Метод пытается разделить данные на заранее указанное количество кластеров с равной дисперсией. Выбирает центроиды будущих кластеров и, минимизируя сумму квадратов расстояний внутри кластера, относит объект к тому или иному кластеру. Показывает хорошие результаты, когда кластеры выпуклые, достаточно удалены друг от друга. Плохо реагирует на кластеры неправильных форм и удлиненные.

## Hierarchical clustering

Иерархическая кластеризация начинается с рассмотрения каждого объекта как отдельного кластера. Затем алгоритм идентифицирует два кластера, которые находятся ближе всего друг к другу, и объединяет их. Этот итерационный процесс продолжается до тех пор, пока все кластеры не будут объединены вместе.

Важным аспектом является метрика измерения расстояния между кластерами. Чаще всего задается евклидовая. Другой аспект – критерий связи между кластерами (откуда рассчитывается расстояние между кластерами). По умолчанию используется Ward, минимизирующий суммы квадратов расстояний каждого объекта от среднего в кластере.

## DBSCAN

Данный алгоритм основан на том, что группирует точки, плотно расположенные друг к другу, и определяет как выбросы точки в областях с маленькой плотностью. Минус метода – невозможность вручную устанавливать количество кластеров. К тому же, DBSCAN помечает удаленные точки как выбросы, что в нашей задаче не является лучшим решением.

В каждом названии или описании кинолокации упоминается тип объекта или его составляющие. Из этого можно сделать вывод, что объекты, определенные DBSCAN’ом, как выбросы, таковыми не являются. В любом случае, для данного кейса важнее посоветовать не совсем подходящий вариант, но сохранить разнообразие предложений и не пропускать объекты.

## Кластеризация данных

В процессе были испробованы все вышеописанные алгоритмы кластеризации. Лучшие результаты показали K-Means и Hierarchical clustering (Agglomerative Clustering). Последний сработал быстрее по времени.

Результат можно видеть на Рисунках 5,6.

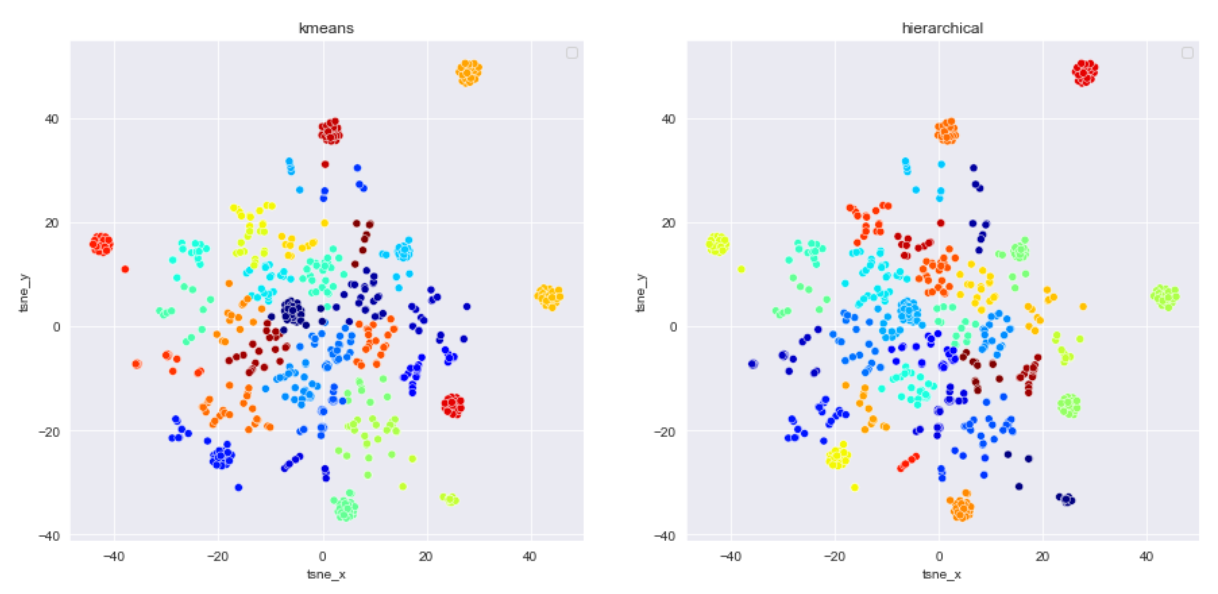


Рисунок 5 Кластеризация (слева - K-Means, справа - Agglomerative Clustering)

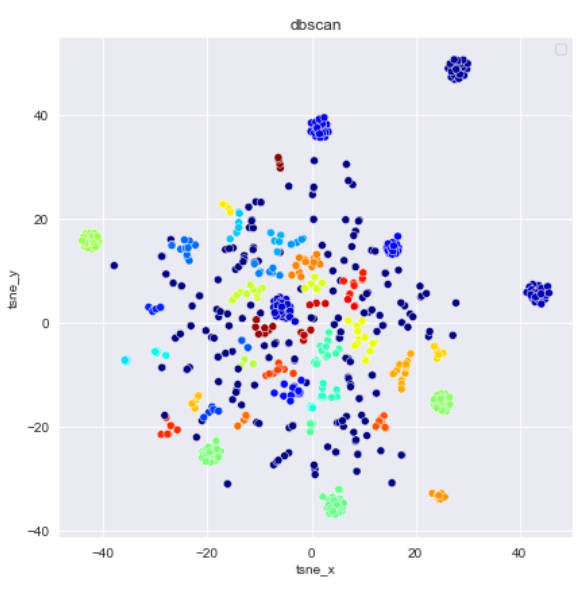
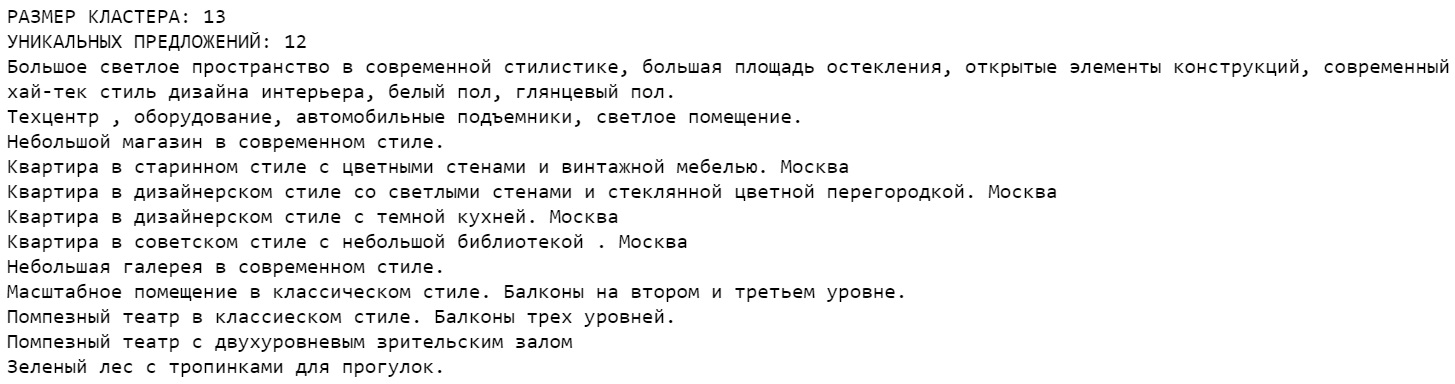


Рисунок 6 Кластеризация (DBSCAN)

Иерархическая кластеризация выделила больше различий внутри большой области, где не было скоплений объектов, поэтому остановимся на ней.

В пределах полученных кластеров наблюдается схожесть между описаниями. Предложения, связанные с аэродромами собраны вместе, предложения по квартирам и лофтам – в другом кластере и т.д.



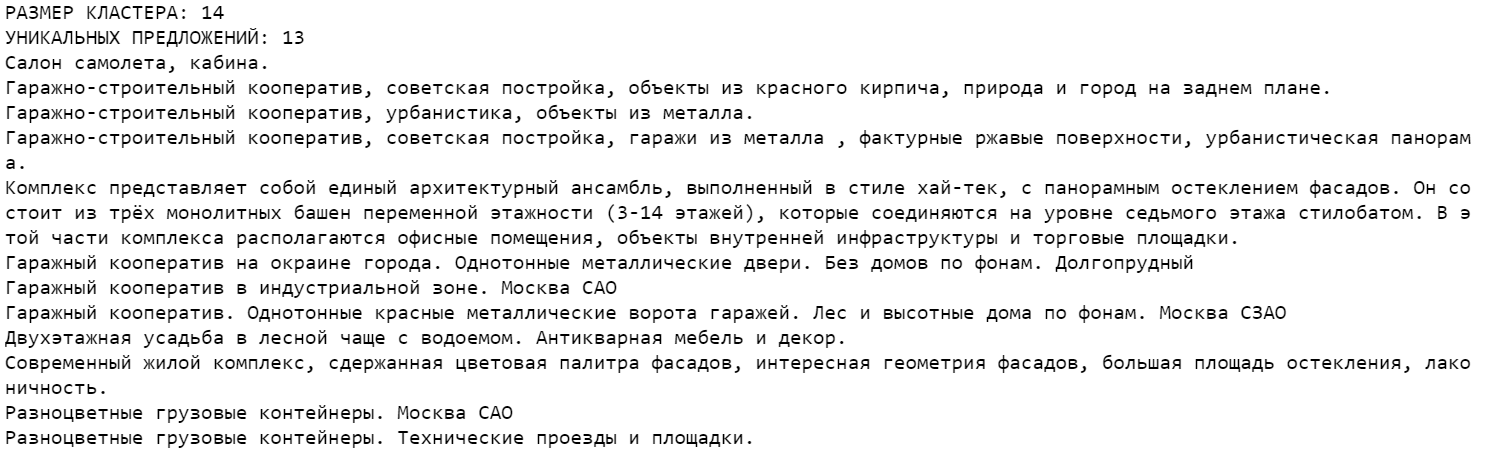


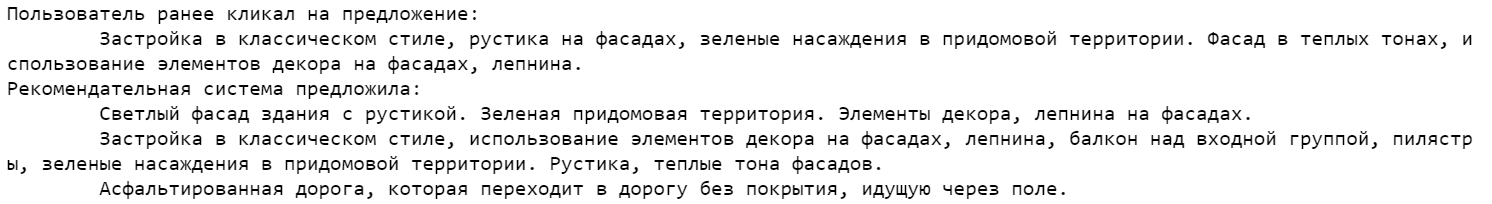
Рисунок 7 Пример полученных кластеров

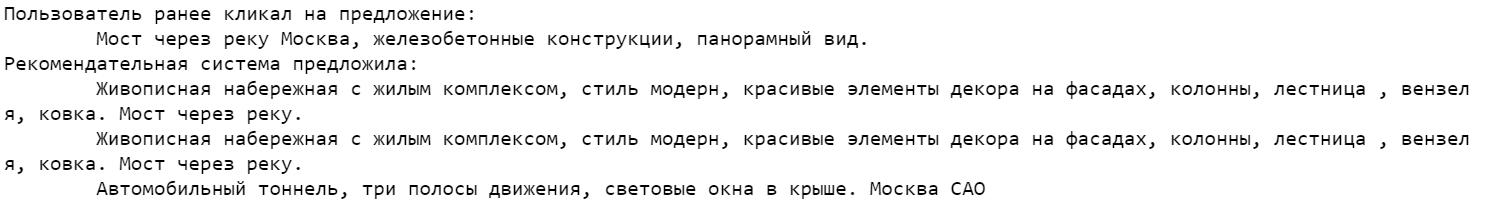
# Рекомендательная система

Как говорилось ранее, у нас не было данных по запросам пользователей, поэтому пример поискового запроса пользователя генерировался вручную - осуществлялся случайный выбор одного из описаний. В части тестирования это не вызовет проблем, так как нам важно понять, насколько похожие предложения сейчас выдает рекомендательная система (как хорошо она устанавливает взаимосвязи между описаниями объектов), что не зависит от профиля пользователя.

Основная часть работы при построении рекомендательной системы заложена в том, чтобы подготовить данные для нее и описать взаимосвязь объектов, на основе которой система будет предлагать пользователю новые объекты. В нашем случае объекты между собой связаны кластерами, внутри которых подразумевается наличие связи между объектами на основе их текстового описания. Близость описаний обеспечивается предобработкой текста в числовые данные благодаря Doc2Vec.

Алгоритму на вход поступает объект, ранее выбранный пользователем (в данном случае случайное описание площадки из имеющихся). Система обращается к кластеру, в котором находится данный запрос и выдает случайный объект из этого же кластера, который и предлагается пользователю в качестве нового.





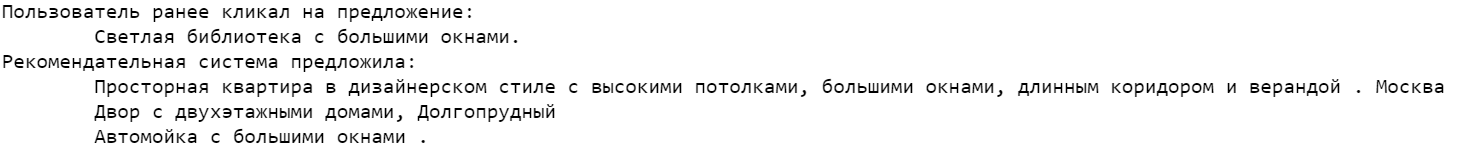
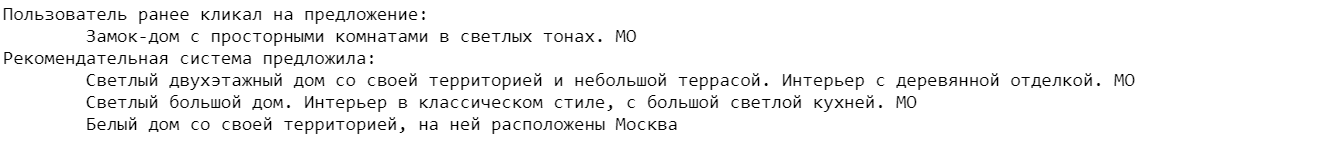
 

Рисунок 8 Примеры рекомендаций

# Заключение

В итоге проделанной работы была построена рекомендательная система, которая предлагает пользователю киноплощадки похожие на те, что он просматривал ранее. Рекомендательная система построена по принципу фильтрации на основе содержания и использует в качестве характеристики объектов их текстовое описание, в том числе название.

Можно задаться вопросом: почему важно пройти все этапы, включая кластеризацию площадок, если можно вместо этого выдавать ближайший запрос на основе Doc2Vec? Фактически можно обойтись и без кластеризации, однако такая рекомендательная система будет использовать только текстовые описания, что не дает возможности в будущем расширить ее и на другие характеристики, такие как местоположение, тэги, изображения и прочее.

Конечно, данная рекомендательная система далека от идеальной. Как минимум, уместно было бы усложнить ее, добавив новые параметры, на которых будут основываться рекомендации. Однако, пока что реализовать подобное не представляется возможным в силу специфики сферы и отсутствия данных в источниках.

Ссылка на репозиторий с кодом работы: <https://github.com/KvindtEva/Course>

## Дальнейшая работа

Для начала стоит отметить, что данные, полученные после парсинга с ресурсов поиска площадок для киносъемок, получились слишком искусственными и для улучшения векторного преобразования строк-описаний, стоит поискать другие датасеты. Сейчас модель Doc2Vec улавливает лишь поверхностные взаимосвязи. Например, повторение словосочетания «большие окна» в описаниях или указания типа пространства «ангар», «лофт». Хотя и это уже хороший результат для данного датасета.

Есть мысль об улучшении Doc2Vec и с другой стороны. Русский язык имеет свою специфичность за счет форм слов, и для нынешней модели слова «краснота» и «красный» воспринимаются как различные, хотя должно быть иначе. Для этого можно поискать предобученные библиотеки русских текстов или алгоритмы обработки русского текста.

В будущем, к уже имеющейся рекомендательной системе стоит добавить дополнительно несколько признаков, на которых будут основываться предложения. Также можно реализовать на сервисе несколько рекомендательных систем, основанных на разных характеристиках объектов, таких как местоположение (математически координаты или текстовое описание), теги, изображения. А с появлением пользовательской базы появится возможность встроить рекомендательные системы по методу коллаборативной фильтрации.

Вышеперечисленные способы улучшения повысят шансы пользователя найти подходящий вариант. Однако на данном этапе, в решении подобных задач мы столкнемся с проблемой примитивности ресурсов по поиску площадок. Все изученные русскоязычные сервисы не обладают достаточной информацией об объектах и пользователях.

Самое главное – внедрение рекомендательной системы в получившийся сервис. К сожалению, эта часть не смогла быть реализована по причине того, что рекомендательная система создавалась параллельно с сервисом в рамках ВКР другой команды.

# Список источников

1. «2021 год станет для российского кинорынка рекордным» | URL: <https://www.vedomosti.ru/media/characters/2020/09/01/838465-dlya-kinorinka-rekordnim>
2. Итоги кинопроката в 2020 году | URL: <https://www.fond-kino.ru/news/itogi-kinoprokata-v-2020-godu/>
3. Как работают рекомендательные системы. Лекция в Яндексе | URL: <https://habr.com/ru/company/yandex/blog/241455/>
4. Рекомендательные системы: идеи, подходы, задачи | URL: <https://habr.com/ru/company/jetinfosystems/blog/453792/>
5. Рекомендательные системы: как помочь пользователю найти то, что ему нужно?| URL: <https://vc.ru/marketing/152926-rekomendatelnye-sistemy-kak-pomoch-polzovatelyu-nayti-to-chto-emu-nuzhno>
6. Векторное представление слов | URL: <https://craftappmobile.com/vector-representation-of-words/>
7. Gensim | URL: <https://radimrehurek.com/gensim/>
8. Gensim Word2Vec Tutorial | URL: <https://www.kaggle.com/pierremegret/gensim-word2vec-tutorial>
9. Современные методы анализа тональности текста | URL: <http://datareview.info/article/sovremennyie-metodyi-analiza-tonalnosti-teksta/>
10. Как работает FastText и где ее применять | URL: <https://sysblok.ru/nlp/kak-rabotaet-fasttext-i-gde-ee-primenjat/>
11. DOC2VEC gensim tutorial | URL: <https://medium.com/@mishra.thedeepak/doc2vec-simple-implementation-example-df2afbbfbad5>
12. A gentle introduction to Doc2Vec | URL: <https://medium.com/wisio/a-gentle-introduction-to-doc2vec-db3e8c0cce5e>
13. Python for NLP: Working with Facebook FastText Library | URL: <https://stackabuse.com/python-for-nlp-working-with-facebook-fasttext-library>
14. Как работает метод главных компонент (PCA) | URL: <https://habr.com/ru/post/304214/>
15. Python Документация | URL: <https://scikit-learn.org/stable/>
16. PCA vs. LDA vs. T-SNE | URL: <https://medium.com/analytics-vidhya/pca-vs-lda-vs-t-sne-lets-understand-the-difference-between-them-22fa6b9be9d0>
17. Понимая PCA, как работают алгоритмы понижения размерности данных | URL: <http://data4.ru/pca>
18. PCA using Python (scikit-learn) | URL: <https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60>
19. An Introduction to t-SNE with Python Example | URL: <https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1>
20. Visualising high-dimensional datasets using PCA and t-SNE in Python | URL: <https://towardsdatascience.com/visualising-high-dimensional-datasets-using-pca-and-t-sne-in-python-8ef87e7915b>
21. Evaluation of UMAP as an alternative to t-SNE for single-cell data | URL: <https://www.biorxiv.org/content/biorxiv/early/2018/04/10/298430.full.pdf>
22. Обзор нового алгоритма уменьшения размерности UMAP | URL: <https://habr.com/ru/company/newprolab/blog/350584/>
23. UMAP Википедия | URL: <https://ru.wikipedia.org/wiki/UMAP>
24. UMAP | URL: <https://umap-learn.readthedocs.io/en/latest/>
25. How to Use UMAP | URL: <https://umap-learn.readthedocs.io/en/latest/basic_usage.html>
26. Linear Discriminant Analysis In Python | URL: <https://towardsdatascience.com/linear-discriminant-analysis-in-python-76b8b17817c2>
27. Linear Discriminant Analysis or LDA in Python | URL: <https://www.mygreatlearning.com/blog/linear-discriminant-analysis-or-lda/#assumptionsoflda>
28. Как уменьшить количество измерений и извлечь из этого пользу | URL: <https://habr.com/ru/post/275273/>