МИНОБРНАУКИ РОССИИ
ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

КУРСОВОЕ ПРОЕКТИРОВАНИЕ

Проектирование системы подбора товара

Б-ИН-21-115074-8847-с. 15.000. КП

Студент
гр. БИН-21-01 _____ П.С. Резниченко

Руководитель,
Доцент,
доктор тех. наук _____ В.М. Гриняк

Науч. консультант,
Ассистент _____ Р.П. Борисов

Владивосток 2025

Аннотация

Актуальность курсового проекта обусловлена потребностью организации ООО «ДНС Технологии» в более точных рекомендациях товаров для конечного потребителя для увеличения конечной прибыли.

В данной курсовом проекте будет рассматриваться проектирование рекомендательной системы, которое состоит из следующих элементов:

– введение в существующие на момент написания работы алгоритмы рекомендаций;

– обзор существующих рекомендательных систем для проектирования своей реализации;

– описание ролей пользователей в рамках проектируемой системы и их возможностей;

– описание методов взаимодействия системы со сторонними ресурсами;

– описание формата данных, которые будет принимать система для последующей обработки;

– описание формата данных, которые будут храниться в системе после обработки;

– общее описание архитектуры проектируемой системы;

– описание алгоритма получения данных для системы из предоставляемого источника;

– описание алгоритма получения оценки на основе текстовых отзывов пользователей;

– описание алгоритма получения рекомендаций на основе выбранных ранее товаров пользователем.

При выполнении курсового проекта было использовано 10 источников. Сам курсовой проект выполнен на 37 страниц, содержит 19 рисунков и 6 диаграмм.

Введение

Рекомендации являются неотъемлемой частью покупок с тех самых времен, когда только начинались товарно-денежные отношения между покупателем и продавцом. В ранних цивилизациях рекомендации часто передавались из уст в уста через знакомых, опытных в какой-либо сфере людей или сами торговцы и ремесленники использовали личные контакты для продвижения своих товаров. В Средние века к процессу предоставления рекомендаций также подключились местные общины и гильдии ремесленников и торговцев через глашатаев. С развитием технологий покупатели стали получать рекомендации сначала через рекламные кампании через рекламные щиты, газеты и прочую типографию, а потом через экраны своих телевизоров, компьютеров и смартфонов.

На момент написания работы достижения в области технологий позволили людям совершать покупки, не выходя из дома, используя разнообразные онлайн-платформы и мобильные приложения. В результате такого прогресса значительно увеличился общий объем покупок, особенно в сфере электронной коммерции.

Однако, это развитие также привнесло ряд новых вызовов, в частности, усложнило процесс создания и предоставления персонализированных рекомендаций для пользователей. Огромное количество данных, собираемых о предпочтениях клиентов, а также разнообразие товаров, представленных на платформах, требует внедрения более совершенных алгоритмов и технологий искусственного интеллекта, чтобы точно предсказать, что именно может заинтересовать каждого конкретного покупателя. Таким образом, хотя прогресс в области онлайн-покупок открыл новые горизонты, он также создал новые проблемы, с которыми нужно эффективно бороться для обеспечения качественного и удобного покупательского опыта.

Для решения вышеописанной проблемы были созданы рекомендательные алгоритмы, на основе которых выстроены современные рекомендательные системы. В рамках курсовой работы будет спроектирована подобная система с использованием методов машинного обучения. Для этого требуется выполнить следующий перечень задач:

- описать принцип работы системы;
- описать структуру данных, которую будет использовать система;
- описать архитектуру системы;
- описать алгоритмы работы системы.

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВВГУ»)

Институт информационных технологий и анализа данных

Кафедра информационных технологий и систем

Индивидуальное задание
На производственную технологическую (проектно-технологическую) практику

Студенту гр. БИН-21 Резниченко Павлу Сергеевичу

1. «Проектирование системы подбора товаров», приказ № «8847-с» от 03.10.2024
2. Срок сдачи работы: 14.01.2025
3. Техническое задание

3.1 Цель

Подбор рекомендованных товаров для покупателя на сайте интернет-магазина

3.2. Технические требования

Система должна принимать данные о выбранных пользователем товарах в виде таблицы формата «.csv». В качестве выходных данных система должна предоставлять файл формата «.json», который содержит рекомендуемые товары для пользователя, чьи данные были отправлены в систему.

4. Курсовой проект в обязательном порядке представляется:

а) пояснительной запиской,

б) графическими материалами.

рекомендуется:

в) компьютерная презентация работы

г) действующий макет устройства или его функционального узла

5. Содержание пояснительной записки.

1) Анализ предметной области
2) Принцип взаимодействия человека с системой
3) Структура данных в системе
4) Архитектура системы
5) Алгоритмы системы

6 Срок сдачи отчета на кафедру: 18.01.2025

Руководитель,
Доцент, канд. физ. мат. наук Гриняк В.М.

Задание получил: Резниченко П.С.

Содержание

1. Анализ предметной области

1.1. Алгоритмы подбора товара

Задача рекомендательной системы – проинформировать пользователя о товаре, который ему может быть наиболее интересен в данный момент времени. Клиент получает информацию, а сервис зарабатывает на предоставлении качественных услуг. Услуги — это не обязательно прямые продажи предлагаемого товара. Сервис также может зарабатывать на комиссионных или просто увеличивать лояльность пользователей, которая потом выливается в рекламные и иные доходы [1].

В зависимости от модели бизнеса рекомендации могут быть его основой, как, например, у TripAdvisor, а могут быть просто удобным дополнительным сервисом (как, например, в каком-нибудь интернет-магазине одежды), призванным улучшить Customer Experience и сделать навигацию по каталогу более удобной.

Персонализация онлайн-маркетинга – очевидный тренд последнего десятилетия, так как предложение, учитывающее как можно больше потребностей и предпочтений покупателя, сделает его счастливее.

Чтобы проиллюстрировать всё многообразие рекомендательных сервисов, можно привести список основных характеристик, с помощью которых можно описать любую рекомендательную систему:

– предмет рекомендации - здесь большое разнообразие, так как тут могут быть товары (Amazon, Ozon), статьи (Arxiv.org), новости (Surfingbird, Дзен), изображения (Imgur, Pinterest), видео (YouTube, Netflix), люди (ВКонтакте, Twitter), музыка (Last.fm, Яндекс.Музыка), плейлисты и прочее. В целом, рекомендовать можно что угодно;

– цель рекомендации – зачем рекомендуется, например, покупка, информирование, обучение, заведение контактов;

– контекст рекомендации – что пользователь в этот момент делает. Например, смотрит товары, слушает музыку, общается с людьми;

– источник рекомендации – кто рекомендует: аудитория (средний рейтинг ресторана в TripAdvisor), схожие по интересам пользователи, экспертное сообщество (бывает, когда речь о сложном товаре, таком, как, например, вино);

– степень персонализации – неперсональные - когда вам рекомендуют то же самое, что всем остальным, что позволяет допускать таргетинг по региону или времени, но не учитывают ваши личные предпочтенъия или когда рекомендации используют данные из вашей текущей сессии, например, когда пользователь посмотрел несколько товаров, и внизу страницы ему предлагаются похожие. Также бывают персональные – те же

рекомендации используют всю доступную информацию о клиенте, в том числе историю его покупок;

– прозрачность - люди больше доверяют рекомендации, если понимают, как именно она была получена. Так меньше риск нарваться на «недобросовестные» системы, продвигающие проплаченный товар или ставящие более дорогие товары выше в рейтинге. Кроме того, хорошая рекомендательная система сама должна уметь бороться с купленными отзывами и накрутками продавцов. Манипуляции кстати бывают и непреднамеренными. Например, когда выходит новый блокбастер, первым делом на него идут фанаты, соответственно, первую пару месяцев рейтинг может быть сильно завышен;

– формат рекомендации - это может быть всплывающее окошко, появляющийся в определенном разделе сайта отсортированный список, лента внизу экрана или что-то еще;

– алгоритмы - несмотря на множество существующих алгоритмов, все они сводятся к нескольким базовым подходам, которые будут описаны далее. К наиболее классическим относятся алгоритмы Summary-based (неперсональные), Content-based (модели основанные на описании товара), Collaborative Filtering (коллаборативная фильтрация), Matrix Factorization (методы основанные на матричном разложении) и некоторые другие.

В центре любой рекомендательной системы находится так называемая матрица предпочтений, пример которой можно увидеть на рисунке 1.1.

	Товар 1	Товар 2	Товар 3	Товар 4	Товар 5
Клиент 1		3		5	
Клиент 2	1		1	1	
Клиент 3	2			3	2
Клиент 4		4			5
Клиент 5	5		2	3	4

Рисунок 1.1. – Пример матрицы предпочтений

Это матрица, по одной из осей которой отложены все клиенты сервиса (Users), а по другой – объекты рекомендации (Items). На пересечении некоторых пар (user, item) данная

матрица заполнена оценками (Ratings) – это известный нам показатель заинтересованности пользователя в данном товаре, выраженный по заданной шкале (например от 1 до 5).

Пользователи обычно оценивают лишь небольшую часть товаров, что есть в каталоге, и задача рекомендательной системы – обобщить эту информацию и предсказать отношение клиента к другим товарам, про которые ничего не известно. Другими словами, нужно заполнить все незаполненные ячейки на картинке выше.

Шаблоны потребления у людей разные, и не обязательно должны рекомендоваться новые товары. Можно показывать повторные позиции, например, для пополнения запаса. По этому принципу выделяют две группы товаров:

– повторяемые: например, шампуни или бритвенные станки, которые нужны всегда;

– неповторямые.: например, книги или фильмы, которые редко приобретают повторно.

Если продукт нельзя явно отнести к одному из классов, имеет смысл определять допустимость повторных покупок индивидуально (кто-то ходит в магазин только ради арахисового масла определенной марки, а кому-то важно попробовать все, что есть в каталоге).

Понятие «интересности» тоже субъективное. Некоторым пользователям нужны вещи только из их любимой категории (conservative recommendations), а кто-то, наоборот, больше откликается на нестандартные товары или группы товаров (risky recommendations). Например, видеохостинг может рекомендовать пользователю только новые серии любимого сериала, а может периодически закидывать ему новые шоу или вообще новые жанры. В идеале стоит выбирать стратегию показа рекомендаций под каждого клиента отдельно, с помощью моделирования категории клиента.

Пользовательские оценки можно получить двумя способами:

– явно (explicit ratings) – пользователь сам ставит рейтинг товару, оставляет отзыв, лайкает страницу;

– неявно (implicit ratings) – пользователь явно свое отношение не выражает, но можно сделать косвенный вывод из его действий: купил товар – значит он ему нравится, долго читал описание – значит есть интерес и т.п.

Конечно, явные предпочтения лучше – пользователь сам говорит о том, что ему понравилось. Однако на практике далеко не все сайты предоставляют возможность явно выражать свой интерес, да и не все пользователи имеют желание это делать. Чаще всего используются сразу оба типа оценок и хорошо дополняют друг друга.

Также важно отличать термины Prediction (предсказание степени интереса) и собственно Recommendation (показ рекомендации). Что и как показывать – это отдельная задача, которая использует полученные на шаге Prediction оценки, но может быть реализована по-разному.

Иногда термин “рекомендация” употребляют в более широком смысле и имеют в виду любую оптимизацию, будь то выборка клиентов для рекламной рассылки, определение оптимальной цены предложения или просто выбор наилучшей стратегии коммуникаций с клиентом.

Для выбора необходимого алгоритма необходимо рассмотреть более подробно неперсонализированные и персонализированные системы. В первом типе потенциальный интерес пользователя определяется просто средним рейтингом товара: «Всем нравится – значит понравится и вам». По этому принципу работает большинство сервисов, когда пользователь не авторизуется в системе, например, тот же TripAdvisor.

Показываться рекомендации могут по-разному – как баннер сбоку от описания товара (Amazon), как результат запроса, отсортированный по определенному параметру (TripAdvisor), или как-то еще.

Рейтинг товара также может изображаться разными способами. Это могут быть звездочки рядом с товаром, количество лайков, разница положительных и отрицательных голосов (как обычно делают на форумах), доля высоких оценок или вообще гистограмма оценок. Гистограммы – наиболее информативный способ, но у них есть один минус – их сложно сравнивать между собой или сортировать, когда нужно вывести товары списком.

Холодный старт – это типичная ситуация, когда ещё не накоплено достаточное количество данных для корректной работы рекомендательной системы (например, когда товар новый или просто его очень редко покупают). Если средний рейтинг посчитан по оценкам всего трёх пользователей (igor92, xyz_111 и oleg_s), такая оценка явно не будет достоверной, и пользователи это понимают. Часто в таких ситуациях рейтинги искусственно корректируют.

Первый способ – показывать не среднее значение, а сглаженное среднее (Damped Mean или Moving average). Смысл таков: при малом количестве оценок отображаемый рейтинг больше тяготеет к некому безопасному «среднему» показателю, а как только набирается достаточное количество новых оценок, «усредняющая» корректировка перестает действовать [2].

Другой подход – рассчитывать по каждому рейтингу интервалы достоверности (confidence Intervals). Математически, чем больше оценок, тем меньше вариация среднего и, значит, больше уверенность в его корректности. А в качестве рейтинга можно выводить,

например, нижнюю границу интервала (Low CI Bound). При этом понятно, что такая система будет достаточно консервативной, с тенденцией к занижению оценок по новым товарам (если, конечно, это не хит).

Поскольку оценки ограничены определенной шкалой (например от 0 до 1), обычный способ расчета интервала достоверности здесь плохо применим: из-за хвостов распределения, уходящих на бесконечность и симметричности самого интервала. Есть альтернативный и более точный способ его посчитать — Wilson Confidence Interval. При этом получаются несимметричные интервалы примерно такого вида, которые можно увидеть на рисунке 1.1.

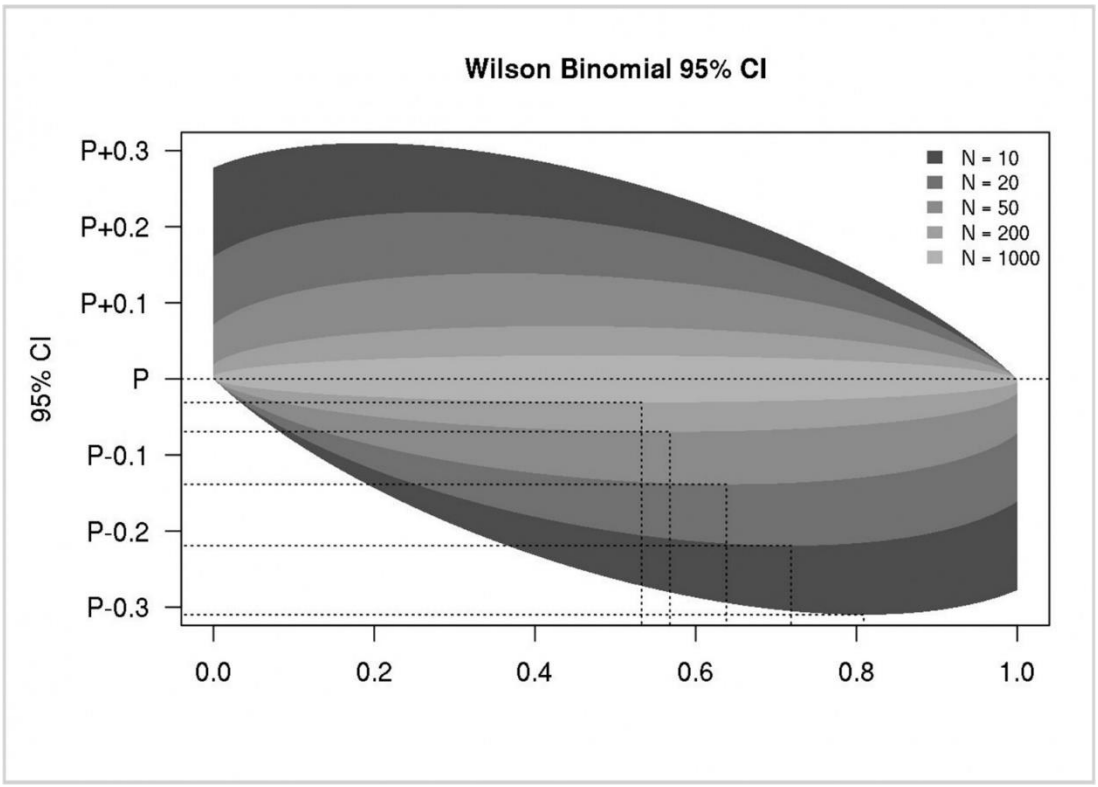

Рисунок 1.1 – Демонстрация интервалов

На рисунке выше по горизонтали отложена оценка среднего значения рейтинга, по вертикали — разброс вокруг среднего значения. Градацией серого выделены различные размеры выборки (чем выборка больше, тем меньше интервал достоверности).

Проблема холодного старта так же актуальна и для неперсонализированных рекомендаций. Общий подход здесь — заменять то, что в данный момент не может быть посчитано, различными эвристиками (например, заменять средним рейтингом, использовать алгоритм попроще, или вообще не использовать товар, пока не соберутся данные).

В некоторых случаях также важно учитывать «свежесть» рекомендации. Это особенно актуально для статей или постов на форумах. Свежие записи должны чаще попадать в топ. Для этого используются корректирующие коэффициенты (damping factors). Ниже приведены формулы для расчета рейтинга статей на медиа сайтах.

Пример расчета рейтинга в журнале «Hacker news» приведен на рисунке 1.2.

$$Rank = \frac{(U-D-1)^{0.8} * P}{T^{1.8}}$$

Рисунок 1.2 – Формула расчета рейтинга в «Hacker news»

На формуле буквы обозначают следующее:

– U (upvotes) – положительные оценки;

– D (downvotes) – отрицательные оценки;

– P (Penalty) — дополнительная корректировка для имплементации иных бизнес-правил

Формула, по которой происходит расчет рейтинга в Reddit указана на рисунке 1.3.

$$Rank = log_{10}(max(1, U - D)) - \frac{|U-D|T}{const}$$

Рисунок 1.3. – Формула расчета рейтинга на сайте «Reddit»

Первое слагаемое оценивает «качество записи», а второе делает поправку на время. В формуле у букв следующее обозначение:

– U = число голосов «за»

– D = число голосов «против»

– T = время записи

Очевидно, что универсальной формулы не существует, и каждый сервис изобретает ту формулу, которая лучше всего решает его задачу – проверяется это эмпирически.

Персональные рекомендации предполагают максимальное использование информации о самом пользователе, в первую очередь о его предыдущих покупках. Одним из первых появился подход content-based filtering. В рамках данного подхода описание товара (content) сопоставляется с интересами пользователя, полученными из его предыдущих оценок. Чем больше товар этим интересам соответствует, тем выше оценивается потенциальная заинтересованность пользователя. Очевидное требование здесь — у всех товаров в каталоге должно быть описание.

Исторически предметом Content-based рекомендаций чаще были товары с неструктурированным описанием: фильмы, книги, статьи. Такими признаками могут быть,

например, текстовые описания, рецензии, состав актеров и прочее. Однако ничто не мешает использовать и обычные числовые или категориальные признаки.

Неструктурированные признаки описываются типичным для текста способом – векторами в пространстве слов (Vector-Space model). Каждый элемент такого вектора – признак, потенциально характеризующий интерес пользователя. Аналогично, продукт – вектор в том же пространстве.

По мере взаимодействия пользователя с системой (скажем, он покупает фильмы), векторные описания приобретенных им товаров объединяются (суммируются и нормализуются) в единый вектор и, таким образом, формируется вектор его интересов. Далее достаточно найти товар, описание которого наиболее близко к вектору интересов, т.е. решить задачу поиска n ближайших соседей.

Не все элементы одинаково значимы: например, союзные слова, очевидно, не несут никакой полезной нагрузки. Поэтому при определении числа совпадающих элементов в двух векторах все измерения нужно предварительно взвешивать по их значимости. Данную задачу решает хорошо известное в Text Mining преобразование TF-IDF, которое назначает больший вес более редким интересам. Совпадение таких интересов имеет большее значение при определении близости двух векторов, чем совпадение популярных. Сама формула принципа изображена на рисунке 1.4.

Рисунок 1.4 – Формула принципа TF-IDF и расшифровка к ней

Принцип TF-IDF здесь в той же мере применим и к обычным номинальным атрибутам, таким, как например, жанр, режиссер, язык. TF — мера значимости атрибута для пользователя, IDF — мера «редкости» атрибута.

Существует целое семейство похожих преобразований (например, BM25 и аналогичные), но содержательно все они повторяют ту же логику, что TF-IDF: редкие атрибуты должны иметь больший вес при сравнении товаров. Рисунок 1.5 иллюстрирует, как именно зависит вес TF-IDF от показателей TF и IDF.

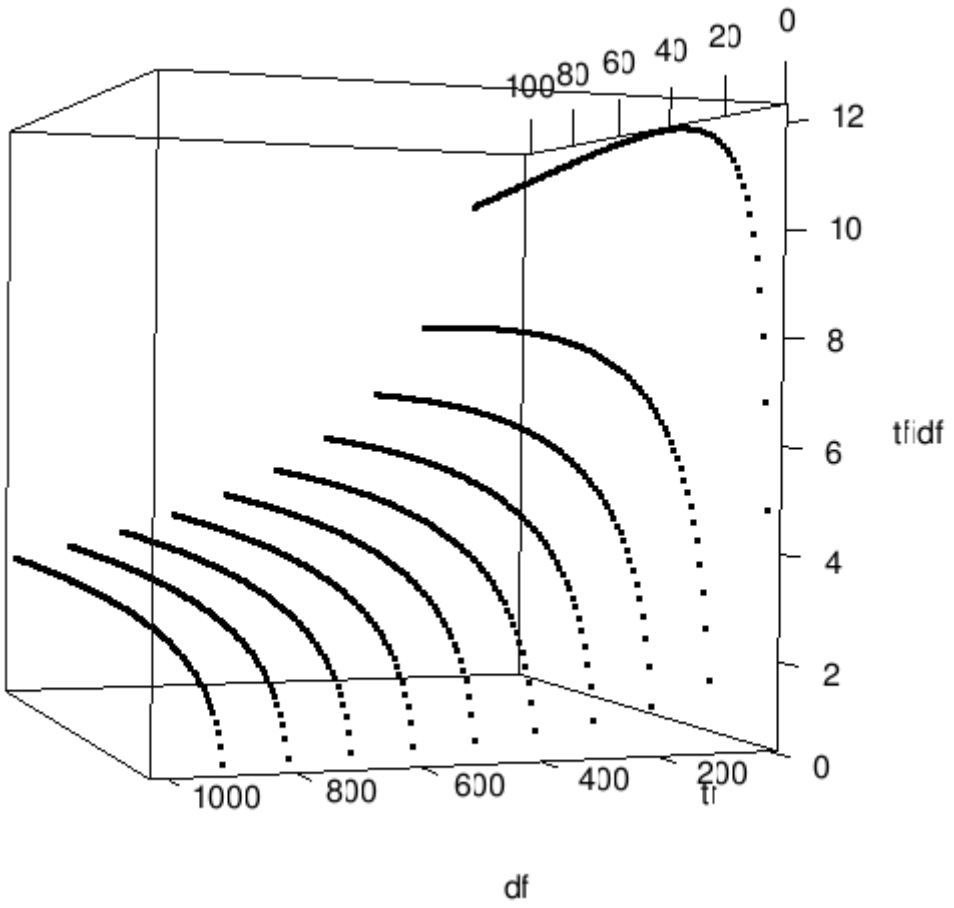

Рисунок 1.5. – Объемная диаграмма зависимостей в TF-IDF

Ближняя горизонтальная ось — это DF: частота атрибута среди всех товаров, дальняя горизонтальная ось — TF: логарифм частоты атрибута у пользователя.

При реализации системы можно учесть следующие моменты:

– При формировании vector-space представления товара вместо отдельных слов можно использовать шинглы или n-граммы (последовательные пары слов, тройки и т.д.). Это сделает модель более детализированной, однако потребуется больше данных для обучения.

– В разных местах описания товара вес ключевых слов может отличаться (например описание фильма может состоять из заголовка, краткого описания и детального описания).

– Описания товара от разных пользователей можно взвешивать по-разному. Например, можем давать больший вес активным пользователям, у которых много оценок.

– Аналогично можно взвешивать и по товару. Чем больше средний рейтинг объекта, тем больше его вес (аналог PageRank).

– Если описание товара допускает ссылки на внешние источники, то можно заморочиться и анализировать также всю связанную с товаром стороннюю информацию.

Видно, что content-based фильтрация почти полностью повторяет механизм query-document matching, используемый в поисковых системах типа Яндекс и Google. Отличие лишь в форме поискового запроса — здесь это вектор, описывающий интересы пользователя, а там — ключевые слова запрашиваемого документа. Когда поисковики стали добавлять персонализацию, различие стерлось еще больше.

В качестве меры близости двух векторов чаще всего используется косинусное расстояние, формула которой расписана на рисунке 1.6.

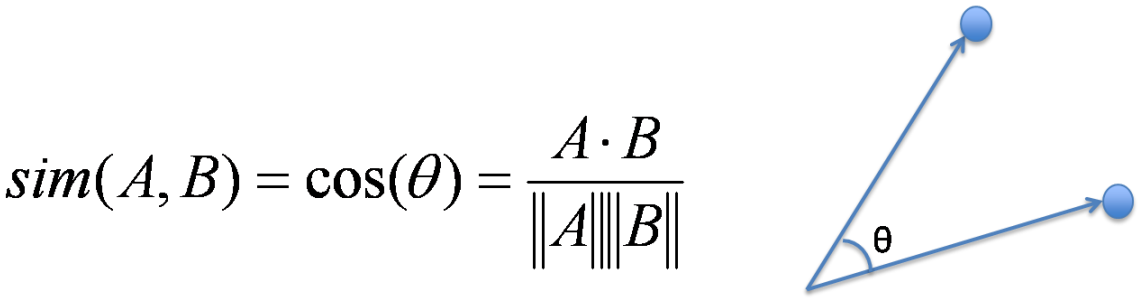

Рисунок 1.6. – Формула косинусного расстояния

При добавлении новой оценки вектор интересов обновляется инкрементально (только по тем элементам, которые изменились). При пересчете имеет смысл давать новым оценкам чуть больше веса, поскольку предпочтения могут меняться.

Класс систем коллаборативной фильтрации (User-based вариант) начал активно развиваться в 90-е годы. В рамках подхода рекомендации генерируются на основании интересов других похожих пользователей. Такие рекомендации являются результатом «коллаборации» множества пользователей. Отсюда и название метода.

Классическая реализация алгоритма основана на принципе k ближайших соседей. На пальцах – для каждого пользователя ищем k наиболее похожих на него (в терминах предпочтений) и дополняем информацию о пользователе известными данными по его соседям. Так, например, если известно, что соседи пользователя по интересам в восторге от фильма «Кровь и бетон», а он сам его по какой-то причине еще не смотрели, это отличный повод предложить ему данный фильм для субботнего просмотра.

На рисунке 1.7 проиллюстрирован принцип работы метода.

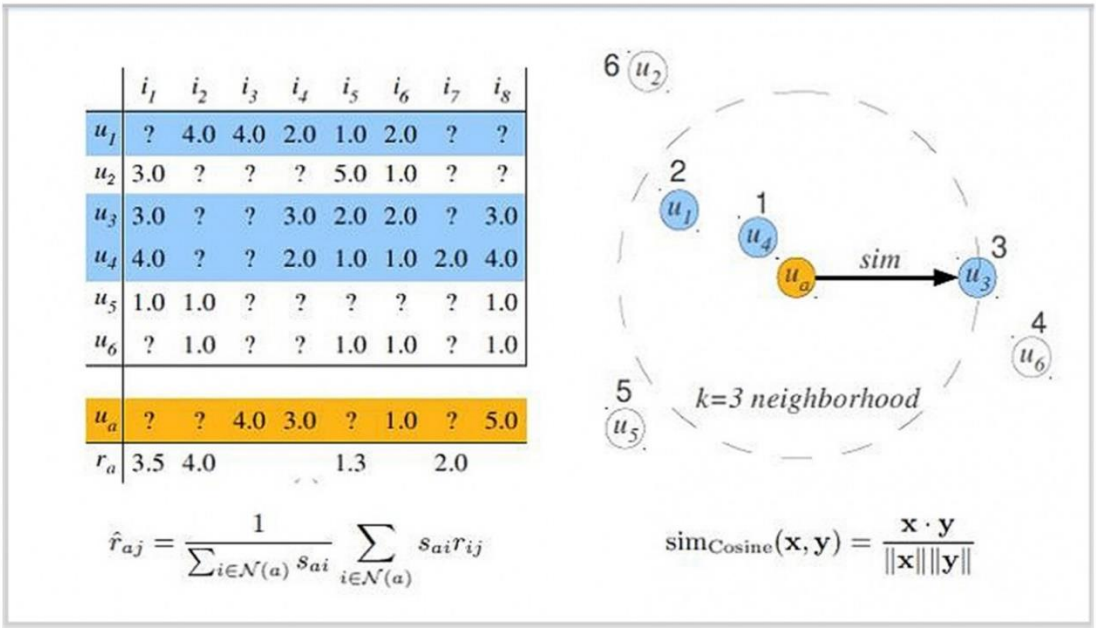

Рисунок 1.7 – Пример работы коллаборативной фильтрации

В матрице предпочтений желтым цветом выделен пользователь, для которого мы хотим определить оценки по новым товарам (знаки вопроса). Синим цветом выделены три его ближайших соседа.

«Похожесть» – в данном случае синоним «корреляции» интересов и может считаться множеством способов (помимо корреляции Пирсона, есть еще косинусное расстояние, есть расстояние Жаккара, расстояние Хэмминга и пр.).

У классической реализации алгоритма есть один явный минус – он плохо применим на практике из-за квадратичной сложности. Действительно, как любой метод ближайшего соседа, он требует расчета всех попарных расстояний между пользователями (а пользователей могут быть миллионы). Нетрудно посчитать, что сложность расчета матрицы расстояний будет $O(n^2m)$, где n — число пользователей, а m — число товаров. При миллионе пользователей для хранения матрицы расстояний в сыром виде, потребуется минимум 4TB.

Данная проблема отчасти может быть решена покупкой высокопроизводительного железа. Но если подходить с умом, то лучше ввести корректировки в алгоритм:

– обновлять расстояния не при каждой покупке, а батчами (например, раз в день),

– не пересчитывать матрицу расстояний полностью, а обновлять ее инкрементально,

– сделать выбор в пользу итеративных и приближенных алгоритмов (например ALS).

Для того чтобы алгоритм был эффективен, важно чтобы выполнялось несколько допущений:

– Вкусы людей не меняются временем (или меняются, но для всех одинаково).

– Если вкусы людей совпадают, то они совпадают во всем.

Например, если два клиента предпочитают одни фильмы, то книги им тоже нравятся одинаковые. Так часто бывает, когда рекомендуемые товары однородны (например, только фильмы). Если это же не так, то у пары клиентов вполне могут совпадать предпочтения в еде, а политические взгляды быть прямо противоположными — здесь алгоритм будет менее эффективным.

Окрестность пользователя в пространстве предпочтений (его соседи), которую мы будем анализировать для генерации новых рекомендаций, можно выбирать по-разному. Мы можем работать вообще со всеми пользователями системы, можем задать некий порог близости, можем выбрать несколько соседей случайным образом или брать n наиболее похожих соседей (это наиболее популярный подход).

Авторы из MovieLens в качестве оптимального количества соседей приводят цифры в 30-50 соседей для фильмов и 25-100 для произвольных рекомендаций. Здесь понятно, что если возьмем слишком много соседей, то получим больше вероятность случайного шума. И наоборот, если взять слишком мало, то получим более точные рекомендации, но меньшее количество товаров можно рекомендовать.

Важный этап подготовки данных — нормализация оценок.

Поскольку все пользователи оценивают по-разному – кто-то всем подряд пятерки ставит, а от кого-то четверки редко дождешься – перед расчетом данные лучше нормализовать, т.е. привести к единой шкале, чтобы алгоритм мог корректно сравнивать их между собой.

Естественно, предсказанную оценку затем нужно будет перевести в исходную шкалу обратным преобразованием (и, если нужно, округлить до ближайшего целого числа).

Нормализовать можно несколькими способами:

– центрированием (mean-centering) — из оценок пользователя просто вычитаем его среднюю оценку (актуально только для небинарных матриц);

– стандартизацией (z-score) — в добавок к центрированию делим оценку ее на стандартное отклонение у пользователя (здесь после обратного преобразования рейтинг может выйти за пределы шкалы (т.е. например, 6 по пятибальной шкале), но такие ситуации довольно редки и решаются просто округлением в сторону ближайшей допустимой оценки);

– двойной стандартизацией — первый раз нормируем оценками пользователя, второй раз — оценками товара.

Если у фильма «Самый лучший фильм» средняя оценка 2.5, а пользователь ей ставит 5, то это сильный фактор, говорящий о том, что такие фильмы ему явно по вкусу.

«Похожесть» или корреляцию предпочтений двух пользователей можно считать разными способами. По сути, необходимо просто сравнить два вектора. Для этого есть несколько популярных способов:

– корреляция Пирсона;

– корреляция Спирмана;

– косинусоидное расстояние;

Корреляция Пирсона — классический коэффициент, который вполне применим и при сравнении векторов. Ее формула изображена на рисунке А.1.

Основной его минус — когда пересечение по оценкам низкое, корреляция может быть высокой просто случайно.

Для борьбы со случайно завышенной корреляцией можно дополнительно умножить на коэффициент 50 / min(50, Rating intersection) или любой другой damping factor, влияние которого уменьшается с ростом числа оценок.

На рисунке А.2 написана формула корреляции Спирмана, основное отличие которой — коэффициент ранговый, т.е. работает не с абсолютными значениями рейтингов, а с их порядковыми номерами. В целом дает результат очень близкий к корреляции Пирсона.

Также, есть еще один классический коэффициент - косинусное расстояние, формула которой приведена на рисунке А.3.

Если приглядеться, косинус угла между стандартизированными векторами — это и есть корреляция Пирсона, одна и та же формула.

Почему косинусное — потому что, если два вектора сонаправлены (т.е. угол между ними нулевой), то косинус угла между ними равен единице. И наоборот, косинус угла между перпендикулярными векторами равен нулю.

Интересное развитие коллаборативного подхода — так называемые Trust-based recommendations, в которых учитывается не только близость людей по интересам, но также их «социальная» близость и степень доверия между ними. Если, например, видно, что на Facebook девушка периодически заходит на страницу с аудиозаписями подруги, значит доверяет её музыкальному вкусу. Следовательно, в рекомендации девушке можно вполне подмешивать новые песни из плейлиста подруги.

Важно, чтобы пользователь доверял рекомендательной системе, а для этого она должна быть проста и понятна. При необходимости всегда должно быть доступно понятное объяснение рекомендации (в англ. терминологии explanation).

В рамках объяснения неплохо показывать оценку товара соседями, по какому именно атрибуту (например, актер или режиссер) было совпадение, а также выводить уверенность системы в оценке (confidence). Чтобы не перегружать интерфейс, можно всю эту информацию вынести в кнопку «Покажи еще», где, например, могут содержаться предложения в духе «Вам может понравиться фильм… поскольку там играет… и ...» или «Пользователи с похожими на ваш музыкальными вкусами оценили альбом… на 4.5 из 5».

Система подбора товара будет работать по принципу неперсонализированных рекомендаций: сначала в систему будут перенесены данные о содержимом корзины покупателя, а затем, основываясь на отзывах других покупателей, пользователь получит рекомендации по приобретении похожего товара.

1.2. Обзор существующих систем

Для обзора существующих систем подбора товара были рассмотрены следующие площадки:

– М.Видео;

– Яндекс.Маркет;

– Ozon;

– Aliexpress.

На сайте «М.Видео» подбор работает следующим образом: система подбирает к каждому товару свои уникальные предложения из смежных или этих же категорий с основным товаром в корзине.

В качестве тестовой выборки были выбраны следующие товары: ноутбук, чайник, смартфон. Рекомендации для данного набора показаны на рисунке 3.1.

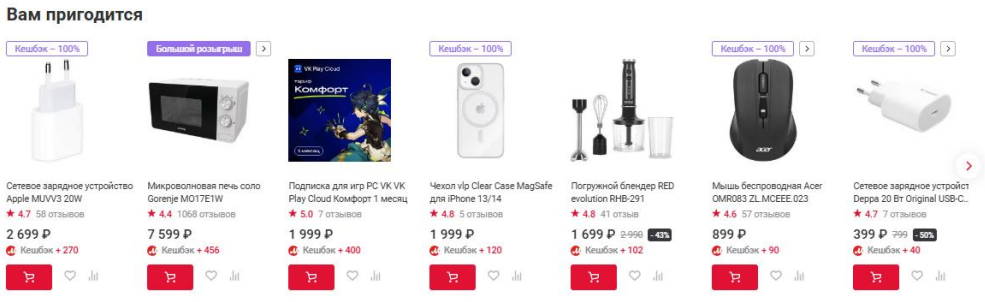

Рисунок 1.1 – результаты подбора на сайте «М.Видео»

Такие позиции как микроволновка и блендер были предоставлены на основе категории товара у чайника. В то время как товары для ноутбука и смартфона были либо взяты из предложенных аксессуаров на основной странице товара, либо основываются на категориях оных.

В маркетплейсе «Ozon» все реализовано следующим образом: система составляет целую ленту из товаров, состоящие в тех же категориях, что и предметы в корзине покупателя.

В качестве тестовой выборки были выбраны следующие товары: спортивный костюм, растворимый кофе и механическая клавиатура. Рекомендации с сайта показаны на рисунке 3.2.

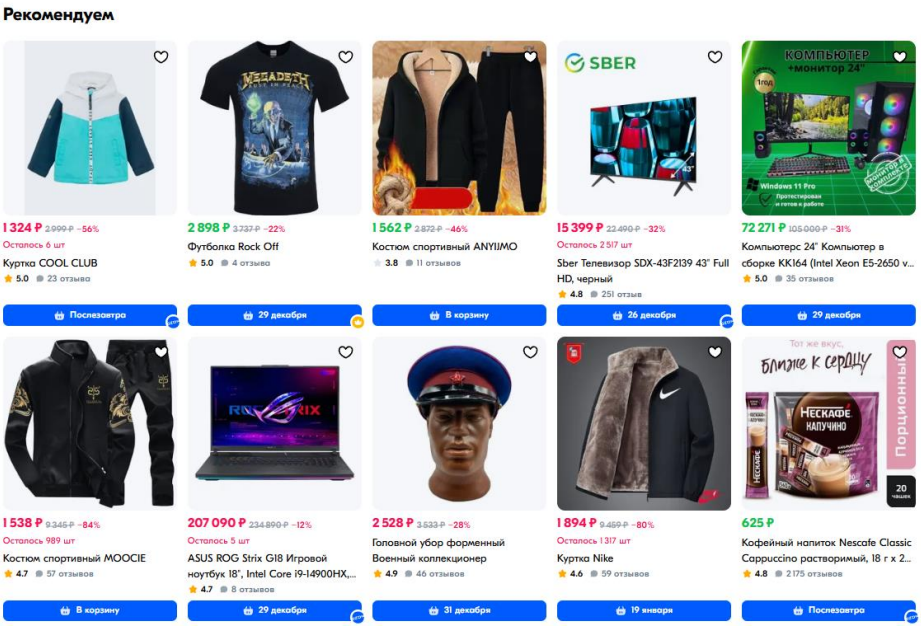

Рисунок 1.2. – результаты подбора на сайте «Ozon»

Большая часть предложений базируется на той категории, в которой находится спортивный костюм. Аналогичным образом был предложен растворимый кофе для растворимого кофе. Компьютер, ноутбук и телевизор были предложены для покупки, т.к. механическая клавиатура является аксессуаром для них.

«Яндекс.Маркет» напротив – добавляет в рекомендованные товары, популярные среди других пользователей на данный момент.

В качестве тестовой выборки были взяты следующие товары: беспроводной игровой контроллер, портативный увлажнитель воздуха. Результаты алгоритма представлены на рисунке 3.3.

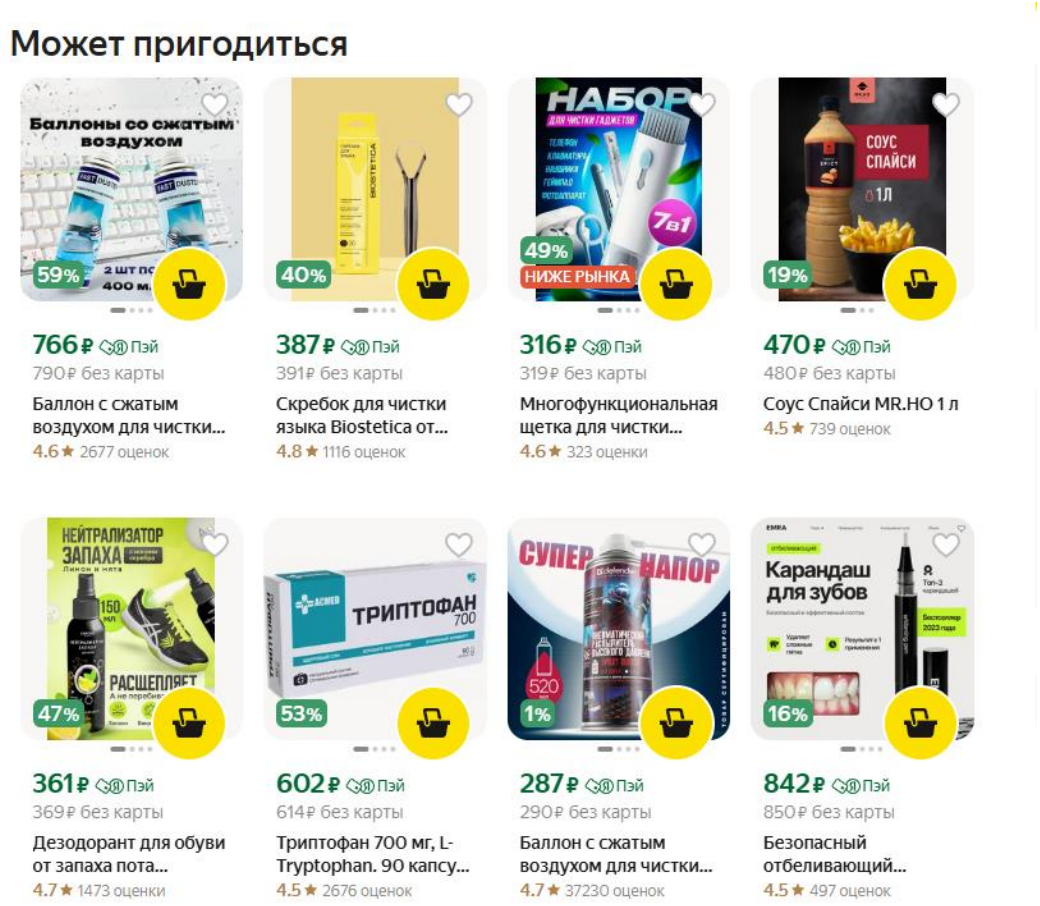

Рисунок 1.3 – Результаты подбора на сайте «Яндекс.Маркет»

Единственные товары, которые относительно могут быть полезны покупателю из данных рекомендованных позиций для тестовой выборки– баллоны с сжатым воздухом, т.к. беспроводные контроллеры, как и любые другие аксессуары для компьютера, могут забиваться пылью.

«Aliexpress» совмещает в себе оба варианта из предыдущих площадок: в рекомендациях находятся популярные товары, как в «Яндекс.Маркете», среди которых в небольшом количестве встречаются товары из той же категории, что и в корзине, как в «Ozon».

В качестве тестовой выборки были выбраны следующие товары: толстовка, статуэтка и блочный конструктор. Результаты подбора показаны на рисунке 3.4.

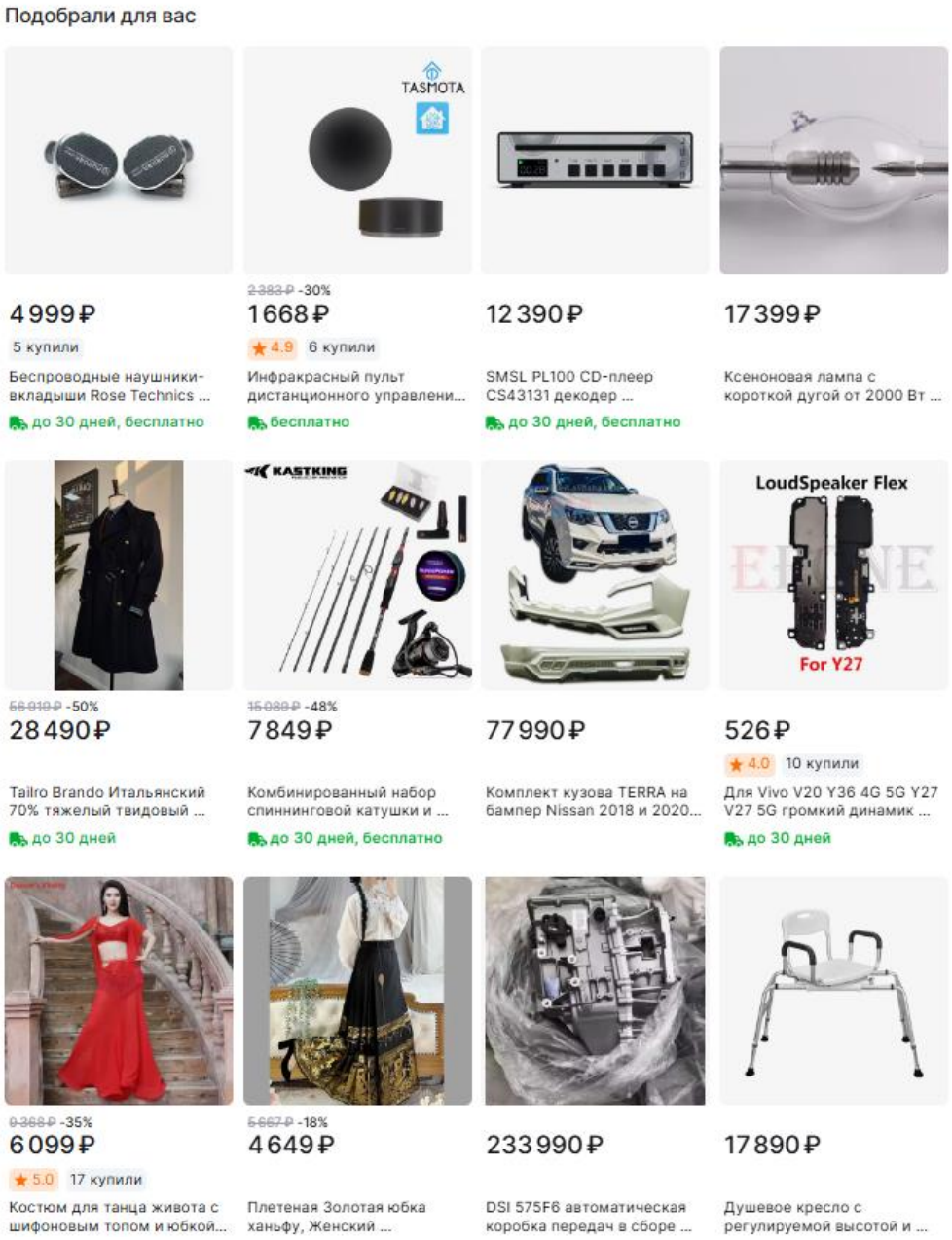

Рисунок 1.4 – Результаты подбора на сайте «Aliexpress»

Единственные товары из рекомендации, которые соотносятся по категории с товарами из корзины – пальто и два платья. Все остальное – это популярные товары у других пользователей.

Исходя из полученных данных, было принято решение проектировать систему, схожую с системой из интернет-магазина «М.Видео», а именно поиск по похожей категории у товара из корзины, т.к. в данном случае она лучше всего подстроиться под возможные потребности пользователя.

2. Взаимодействие с системой

2.1. Роли пользователей

В системе будут следующие роли: «покупатель» и «администратор».

Для описания возможностей описанных выше ролей будут использоваться диаграммы вариантов использования (Use Case Diagrams) - диаграмма, отражающая отношения между актерами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне [3].

Основными элементами данной диаграммы являются [4]:

– участник - это множество логически связанных ролей, исполняемых при взаимодействии с прецедентами или сущностями (система, подсистема или класс). Участником может быть человек или другая система, подсистема или класс, которые представляют нечто вне сущности. Графически участник изображается «человечком»;

– прецедент - описание множества последовательных событий (включая варианты), выполняемых системой, которые приводят к наблюдаемому участником результату. Прецедент представляет поведение сущности, описывая взаимодействие между участниками и системой. Прецедент не показывает, «как» достигается некоторый результат, а только «что» именно выполняется. Прецеденты обозначаются очень простым образом - в виде эллипса, внутри которого указано его название.

Возможности «покупателя» отображены на рисунке Б.1.

Возможность «Формировать корзину» означает, что «покупатель» составит корзину товаров, которых он хочет приобрести, и на основе которой будет формироваться список рекомендуемых товаров.

Возможность «Получать рекомендации» означает, что «покупатель» получит список рекомендованных к покупке товаров, основанный на сформированной им корзине.

Возможности «покупателя» отображены на рисунке Б.2.

Возможность «Загружать данные для тренировки» означает, что «администратор» добавит данные в систему, на которых она сможет натренировать модель, которая поможет выдавать полезные для «пользователя» товары.

Возможность «Редактировать список доступных товаров» означает, что «администратор» добавит новые товары, которые были добавлены в основную базу данных сайта, и удалит те позиции, которые больше не продаются в магазине.

2.2. Программный интерфейс

Для того, чтобы сторонний сайт мог взаимодействовать с рекомендательной системой, у последней есть свой API - набор правил и протоколов, который позволяет разным программам взаимодействовать друг с другом. API определяет методы и структуры данных, которые могут быть использованы для обмена информацией и выполнения операций между различными программами или компонентами ПО [5].

API используется для следующих целей:

- взаимодействие с внешними сервисами;
- расширение функциональности;
- интеграция с аппаратным обеспечением;
- обмен данными.

Само API системы составлено по архитектуре REST API, позволяющему обмениваться сообщениями без сохранения состояния. Каждое сообщение самодостаточное и содержит всю информацию, необходимую для его обработки. Сервер не хранит результаты предыдущих сессий с клиентскими приложениями. Это обеспечивает гибкость и масштабируемость серверной части, позволяет поддерживать асинхронные взаимодействия и реализовывать алгоритмы обработки любой сложности. Кроме того, такой формат взаимодействия является универсальным — он не зависит от технологий, используемых на клиенте и на сервере, и не привязывает разработчиков к определенному провайдеру [6].

При данной архитектуре используются следующие методы:

- GET — получение информации об объекте (ресурсе).
- POST — создание нового объекта (ресурса).
- PUT — полная замена объекта (ресурса) на обновленную версию.
- PATCH — частичное изменение объекта (ресурса).
- DELETE — удаление информации об объекте (ресурсе).

В самой же системе будут следующие методы:

«get_rec» - GET-метод, выдающий список рекомендованных товаров в формате json. Для использования требует json-файл, где перечислены все товары в корзине покупателя;

«add_good» - POST-метод, добавляющий товары в рекомендательную систему. Для использования требует json-файл, где перечислены код товара и его характеристики;

«remove_good» - PATCH-метод, который переносит товары, которые более не продаются, в таблицу-корзину. Для использования требует массив из кодов товаров.

3. Структура данных

3.1. Первоначальный формат данных

Изначально, данные представлены в формате «.xlsx» - формат документов Microsoft Excel, который был представлен корпорацией Майкрософт в выпуске Microsoft Office 2007, и использующий стандарт Open XML [7].

Сама таблица с данными отображена на рисунке В.1.

В этой таблице содержатся данные о товарах в магазине, а также отзывы пользователей о них. Данная таблица состоит из следующих столбцов:

- «product_guid» - код продукта
- «product_name» - название продукта
- «cat1_guid» - код основной категории
- «cat1» - название основной категории
- «cat2_guid» - код вторичной категории
- «cat2» - название вторичной категории
- «cat3_guid» - код третичной категории
- «cat3» - название третичной категории
- «cat4_guid» - код вспомогательной категории
- «cat4» - название вспомогательной категории
- «specs» - массив из характеристик товара
- «price» - цена продукта
- «review_json» - массив из json-объектов, хранящих отзывы покупателей

3.2. Хранение данных в системе

Данные для рекомендательной системы, которые были получены от «администратора», после обработки разделены на две разные таблицы: «df» и «df_reviews».

Структура таблицы «df» отображена на рисунке 3.1.

	product_guid	cat1	cat2	cat3	cat4	Price	mean_grade	reviews_number	months_of_usage	grade
0	1854	1	23	17	162	13999	4.325301	83	1	5
1	1854	1	23	17	162	13999	4.325301	83	1	3
2	1854	1	23	17	162	13999	4.325301	83	1	5
3	1854	1	23	17	162	13999	4.325301	83	1	5
4	1854	1	23	17	162	13999	4.325301	83	1	5
...	...	...	...	...	...	...	...	...	...	...
101183	2339	1	7	20	212	61499	4.600000	10	1	5
101184	2339	1	7	20	212	61499	4.600000	10	1	5
101185	2339	1	7	20	212	61499	4.600000	10	1	5
101186	2339	1	7	20	212	61499	4.600000	10	1	5
101187	2339	1	7	20	212	61499	4.600000	10	2	5

Рисунок 3.1 – Структура таблицы «df»

Данная таблица предназначена для тренировки модели и содержит в себе все оценки для всех товаров. В ней имеются следующие столбцы:

– «product_guid» - код продукта;

– «cat1» - код основной категории;

– «cat2» - код вторичной категории;

– «cat3» - код третичной категории;

– «cat4» - код вспомогательной категории;

– «price» - цена продукта;

– «reviews_number» - общее количество отзывов;

– «months_of_usage» - количество месяцев использования продукта перед написанием отзыва;

– «grade» - оценка товара из отзыва по шкале от 0 до 5, где 0 – наихудшее качество продукта, а 5 – наилучшее.

Структура таблицы «df_reviews» отображена на рисунке 3.2.

	update_stamp	plus	minus	comment	months_of_usage	grade	product_guid
0	2024-01-01	Монитор за свои деньги очень достойный.\n144HZ...	Нету	За свою цену монитор очень классный. Не пожале...	1	5	7697de0b-330f-11ed-901a-00155d8ed20b
1	2024-01-01	После старого ноута изображение норм	Засветы блин	Если вы собираетесь играть в темные игры без с...	1	3	7697de0b-330f-11ed-901a-00155d8ed20b
2	2024-01-02	За такие деньги кладут большой комплект	Есть сильные засветы	Лучший лоник за свои деньги	1	5	7697de0b-330f-11ed-901a-00155d8ed20b
3	2024-01-03	03.01.2024 Сегодня стал обладателем данного мо...	Ножки монитора через чур большие как по мне за...	По сравнению со стареньким 60 гц\r\nЭтот небо ...	1	5	7697de0b-330f-11ed-901a-00155d8ed20b
4	2024-01-03	Цена и качество монитора	нету	Хороший монитор по хорошей цене	1	5	7697de0b-330f-11ed-901a-00155d8ed20b
...	...	...	...	...	...	...	...
101183	2024-07-19	Производительность\nСвежий сокет	цена(субъективно)\nочень горячий	Вау эффект после перехода с i7-3770k. Браузер ...	1	5	94e74a2e-3944-11ed-9021-00155d8ed20b
101184	2024-08-07	Дикая производительность	Ак620 не хватает, долбится в тестах в 89, плюс...	Берите помощнее охлад + оставляйте вариант сто...	1	5	94e74a2e-3944-11ed-9021-00155d8ed20b
101185	2024-09-06	16 ядер	Их нет\nОсобенности:\n- греется, тдп 170в\n- с...	Не дождался в РФ модель Ryzen 9 9950x хотя он ...	1	5	94e74a2e-3944-11ed-9021-00155d8ed20b
101186	2024-09-11	Очень производительный. Брал для разработки, п...	Горячеват, пришлось возиться с ryzen master дл...	Очень ждал 9950x, не дождался, решил взять это...	1	5	94e74a2e-3944-11ed-9021-00155d8ed20b
101187	2024-09-25	Производительность отличная, поддерживаемые ин...	95 градусов без доработок в нагрузке это норма...	Спустя полгода использования унёс его на свида...	2	5	94e74a2e-3944-11ed-9021-00155d8ed20b

Рисунок 3.2 – Структура таблицы «df_reviews»

Данная таблица предназначена для обработки текста отзывов, чтобы улучшить рекомендации от системы и содержит в себе текстовые отзывы товаров. В ней имеются следующие столбцы:

– «update_stamp» - дата обновления отзыва;

– «plus» - выделенные достоинства товара;

– «minus» - выделенные недостатки товара;

– «comment» - дополнительный комментарий по товару;

– «months_of_usage» - количество месяцев использования продукта перед написанием отзыва;

– «grade» - оценка товара из отзыва по шкале от 0 до 5, где 0 – наихудшее качество продукта, а 5 – наилучшее;

– «product_guid» - код продукта.

Сам же алгоритм передает данные в формате json – в текстовом формате обмена данными, основанный на JavaScript и разработанный Дугласом Крокфордом. Как и многие другие текстовые форматы, JSON легко читается людьми.

Несмотря на происхождение от JavaScript (точнее, от подмножества языка стандарта ECMA-262 1999 года), формат считается независимым от языка и может использоваться практически с любым языком программирования. Для многих языков существует готовый код для создания и обработки данных в формате JSON [8].

Данный файл имеет следующую структуру:

– «prod_id» - код продукта в виде набора символов;

– «rank» - показатель вероятности того, что предложенный товар понравиться пользователю.

4. Архитектура системы

Для демонстрации архитектуры системы используется диаграмма контейнеров [9].

Контейнеры здесь не означают обязательно докер-контейнеры. Контейнер — это любой развертываемый объект или хранилище данных с точки зрения C4. Это может быть мобильное приложение, веб-сайт, виртуальная машина, докер-контейнер, база данных или хранилище объектов; все, что можно развернуть. Пример данной диаграмм показан на рисунке Г.1.

Данная диаграмма рисуется следующим образом:

– определяется список сущностей: микросервисы, хранилища, внешние сервисы, после чего их помещают на диаграмму;

– добавление комментариев о назначении каждого компонента и технологии, которую он реализует;

– добавление соединений со стрелками и значимых меток к каждой стрелке;

– подбор необходимой цвет схемы;

– создание легенду, объясняющую каждый элемент диаграммы.

Архитектура системы показана на рисунке Д.1.

Система состоит из следующих компонентов:

– семантический блок – компонент, выставляющий свою оценку на основе текста пользователя и передающий её тренеру;

– передатчик – компонент, который предоставляет тренеру данные, сохраненные в таблице и в корзине покупателя, а также текст отзыва сематическому блоку;

– тренер – компонент, который изучает предоставленные ему данные и тренирует модель, которая поможет анализатору в предоставлении рекомендации

– анализатор – компонент, который на основе натренированной модели выдает список предложений сайту.

5. Алгоритмы системы

5.1. Получение данных для системы

Алгоритм получения данных для системы изображен на рисунке 5.1.

Рисунок 5.1. – Блок-схема получения данных для системы

Сначала, администратор подгружает данные в систему в виде таблицы формате «.xlsx».

Далее, передатчик в системе отделяет полученные данные на отзывы и оценки, после чего делает их векторизацию – преобразование текстовых данных в числовые для дальнейшей работы системы с данными [10].

Наконец, передатчик отправляет «текст» отзыва в семантический блок, а оставшиеся данные тренеру.

5.2. Получение оценки на основе отзывов

Алгоритм получения оценки на основе отзывов изображен на рисунке 5.2.

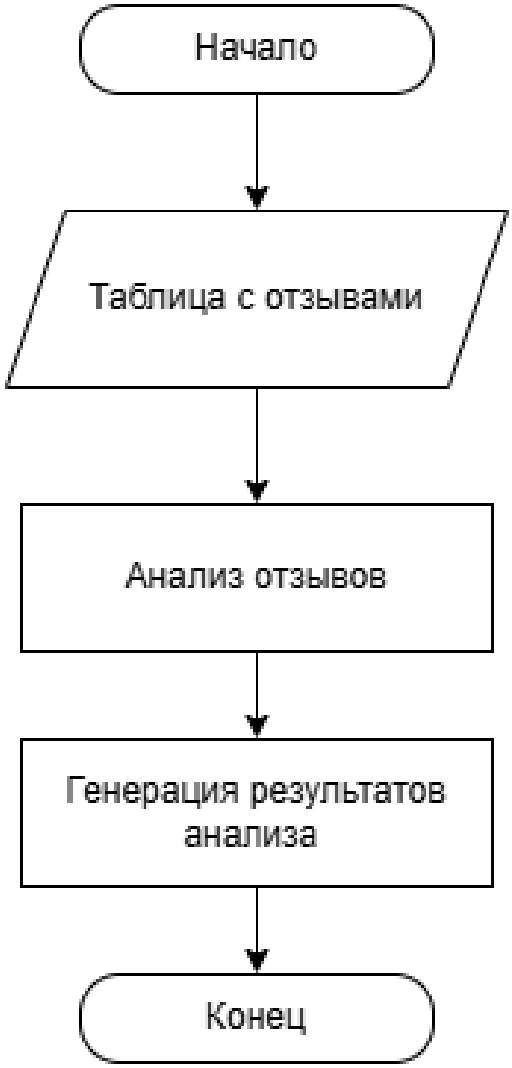

Рисунок 5.2 – Блок-схема для получения оценки на основе отзывов

Сначала семантический блок анализирует полученные отзывы, после чего генерирует таблицу, состоящую из следующих столбцов:

– «id_review» - номер отзыва в виде набора символов;

– «is_positive» - булева переменная. Если «true», то отзыв положительный, иначе он – отрицательный.

После генерации блок отправляет результат обработки тренеру.

5.3. Выдача рекомендуемых покупок

Алгоритм получения оценки на основе отзывов изображен на рисунке 5.3.

Рисунок 5.3 - Блок-схема для выдачи рекомендуемых покупок

Сначала, тренер, на основе полученных данных, тренирует модель предсказаний. После этого модель отправляется анализатору для использования.

Анализатор, в свою очередь, используя модель модели и данные о корзине пользователя, предсказывает какие товары могут понравиться пользователю. Результаты анализа отправляются по запросу в виде json-файла.

Заключение

В рамках курсовой работы для проектирования рекомендательной системы было выполнено следующее:

- описаны роли пользователей в системе;
- описана структура данных, которую будет использовать система;
- описана архитектуру системы;
- описаны алгоритмы работы системы.

Помимо вышеописанного также было сделано следующее:

- отображены существующие на момент написания работы алгоритмы рекомендаций;
- был проведен краткий обзор существующих рекомендательных систем для проектирования своей реализации;
- были описаны методы взаимодействия системы со сторонними ресурсами;
- описаны форматы данных, которые будет принимать система для последующей обработки и которые будут храниться в системе после обработки;
- было приведено общее описание архитектуры проектируемой системы;
- описаны алгоритмы получения данных для системы из предоставляемого источника, оценки на основе текстовых отзывов пользователей и рекомендаций на основе выбранных ранее товаров пользователем.

Результаты проведенной работы позволят создать прототип системы, который в будущем можно интегрировать на сайт магазина или цифровой маркетплейс.

Список использованных источников

1. Анатомия рекомендательных систем. Часть первая [Электронный ресурс] // Хабр – Режим доступа: https://habr.com/ru/companies/lanit/articles/420499/ (дата обращения 06.01.2025);

2. Скользящая средняя [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D0%BE%D0%BB%D1%8C%D0%B7%D1%8F%D1%89%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%BD%D1%8F%D1%8F (дата обращения 06.01.2025);

3. Проектирование Use Case диаграммы. Определение функциональных возможностей системы [Электронный ресурс] // WorldSkills Russia – Режим доступа: https://nationalteam.worldskills.ru/skills/proektirovanie-use-case-diagrammy-opredelenie-funktsionalnykh-vozmozhnostey-sistemy/ (дата обращения 06.01.2025);

4. Диаграмма вариантов использования (UseCase diagram) [Электронный ресурс] // FlexBerry platform – Режим доступа: https://flexberry.github.io/ru/fd_use-case-diagram.html (дата обращения 06.01.2025);

5. API от А до Я (теория и практика) [Электронный ресурс] // Хабр – Режим доступа: https://habr.com/ru/articles/768752/ (дата обращения 06.01.2025);

6. REST API: для чего нужен и как работает [Электронный ресурс] // Yandex Cloud – Режим доступа: https://inlnk.ru/KeROok (дата обращения 06.01.2025);

7. Форматы файлов электронных таблиц .XLSX [Электронный ресурс] // Fileformat – Режим доступа: https://docs.fileformat.com/ru/spreadsheet/xlsx/ (дата обращения 06.01.2025);

8. JSON [Электронный ресурс] // Википедия – Режим доступа: https://ru.wikipedia.org/wiki/JSON (дата обращения 06.01.2025);

9. 5 диаграмм, необходимых для документирования архитектуры решений [Электронный ресурс] // Хабр – Режим доступа: https://habr.com/ru/companies/epam_systems/articles/538018/ (дата обращения 06.01.2025);

10. Краткий обзор техник векторизации в NLP [Электронный ресурс] // Хабр – Режим доступа: https://habr.com/ru/articles/778048/ (дата обращения 06.01.2025).

Приложение А

$$\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}$$

Рисунок А.1 – Формула корреляции Пирсона

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}.$$

Рисунок А.2 – Формула корреляции Спирмана

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

Рисунок А.3 - Формула косинусного расстояния

Приложение Б

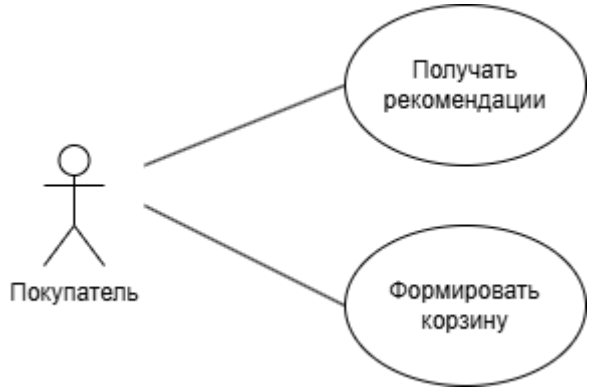

Рисунок Б.1. – возможности роли «покупатель»

Рисунок Б.2. – возможности роли «администратор»

Приложение В

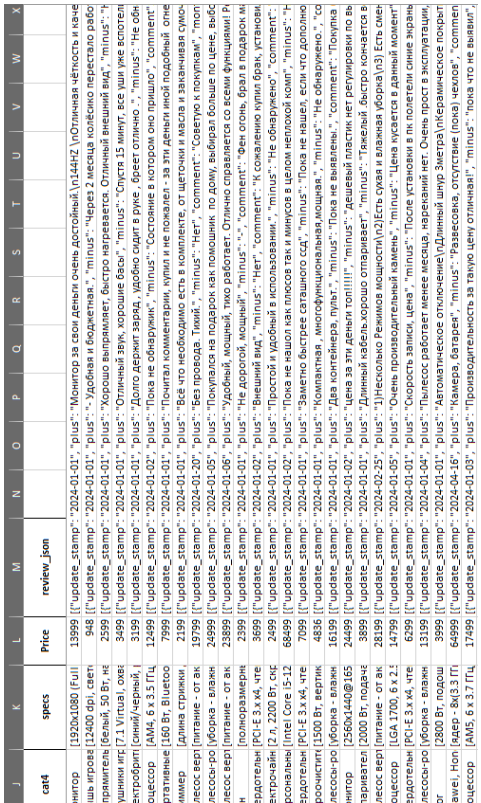

Рисунок В.1 – Таблица с данными

Приложение Г

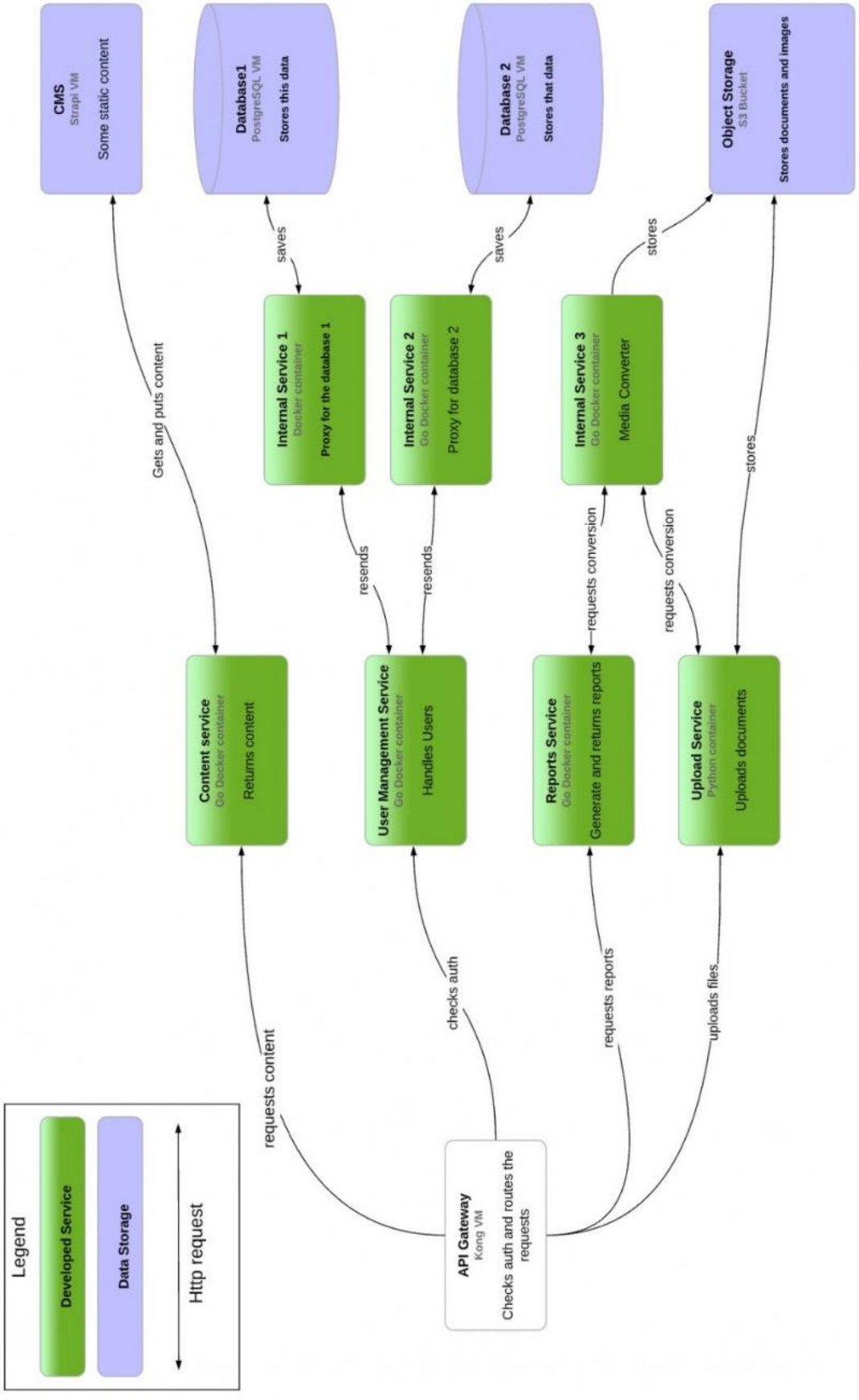

Рисунок Г.1 – Пример диаграммы контейнеров

Приложение Д

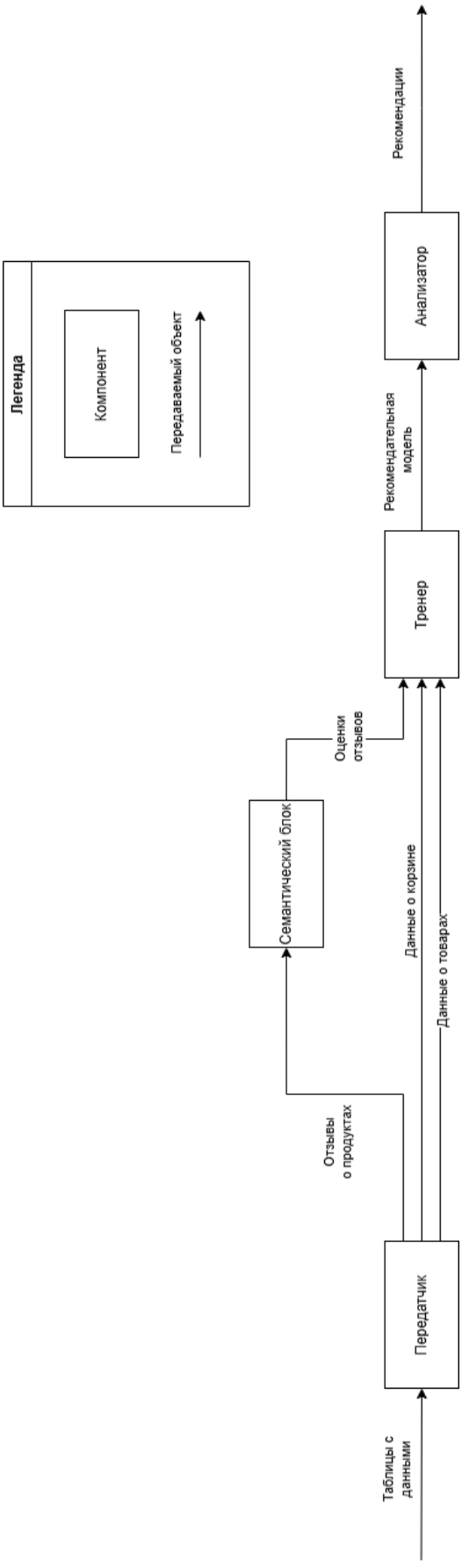

Рисунок Г.1. – Схема архитектуры системы