**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)»

**Институт (Филиал)** № 8 «Компьютерные науки и прикладная математика» **Кафедра**  806

**Группа** М8О-214М-23  **Направление подготовки** 02.04.02 Фундаментальная информатика и информационные технологии

**Профиль**  Информатика и компьютерные науки

**Квалификация**  **магистр**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**МАГИСТРА**

На тему: Рекомендательная система с эвристикой, оптимизирующей маршрут

Автор ВКРБ Шекунов Михаил Андреевич (\_\_\_\_\_\_\_\_\_\_)

(фамилия, имя, отчество полностью)

Руководитель Лунева Светлана Юрьевна (\_\_\_\_\_\_\_\_\_\_)

(фамилия, имя, отчество полностью)

Консультант (\_\_\_\_\_\_\_\_\_\_)

(фамилия, имя, отчество полностью)

Консультант (\_\_\_\_\_\_\_\_\_\_)

(фамилия, имя, отчество полностью)

Рецензент (\_\_\_\_\_\_\_\_\_\_)

(фамилия, имя, отчество полностью)

**К защите допустить**

Заведующий кафедрой 806 Крылов Сергей Сергеевич (\_\_\_\_\_\_\_\_\_)

(№ каф) (фамилия, имя, отчество полностью)

\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025г.

Москва 2025

**РЕФЕРАТ**

Выпускная квалификационная работа бакалавра состоит из 00 страниц, 00 рисунков, 00 таблиц, 00 использованных источников, 00 приложений.

Текст реферата должен отражать:

* + объект исследования или разработки;
  + цель работы;
  + методы или методологию проведения работы;­­­
  + результаты работы и их новизну;
  + область применения результатов;
  + рекомендации по внедрению или итоги внедрения результатов;
  + экономическую эффективность или значимость работы;
  + прогнозные предположения о развитии объекта исследования.

Если отчет не содержит сведений по какой-либо из перечисленных структурных частей реферата, то в тексте реферата она опускается, при этом последовательность изложения сохраняется.

Оптимальный объем текста реферата - 850 печатных знаков, но не более одной страницы машинописного текста.

В общее количество таблиц и рисунков входят и таблицы, и рисунки приложений.

СОДЕРЖАНИЕ

[ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 4](#_Toc198039344)

[ВВЕДЕНИЕ 5](#_Toc198039345)

[1 ТЕОРИЯ 10](#_Toc198039346)

[1.1 Потребности в разработке гибридной рекомендательной системы на базе стохастической оптимизации для культурных учреждений 10](#_Toc198039347)

[1.2 Анализ технических возможностей для реализации алгоритмов 11](#_Toc198039348)

[1.3 Базы данных 11](#_Toc198039349)

[1.4 Поиск существующих рекомендательных систем 12](#_Toc198039350)

[1.5 Сопоставление имеющихся данных и существующих алгоритмов 12](#_Toc198039351)

[1.6 Коллаборативная фильтрация 14](#_Toc198039352)

[1.7 Система на основе схожести объектов 16](#_Toc198039353)

[1.8 Постановка задачи эвристики 19](#_Toc198039354)

[1.9 Метаэвристика HSATS 19](#_Toc198039355)

[1.11 Сравнение HSATS с конкурентом 20](#_Toc198039356)

[1.12 Гибридная рекомендательная система 22](#_Toc198039357)

[1.13 Слияние результатов гибридной системы с результатами стохастической оптимизации 23](#_Toc198039358)

[2 ПРАКТИЧЕСКАЯ ЧАСТЬ 25](#_Toc198039359)

[2.1 Используемый язык программирования 25](#_Toc198039360)

[2.2 Используемые библиотеки и модули 26](#_Toc198039361)

[2.3 Реализация коллаборативной фильтрации 27](#_Toc198039362)

[2.4 Реализация рекомендательной системы, основанной на схожести объектов 30](#_Toc198039363)

[2.5 Реализация объединения данных всех рекомендательных систем 32](#_Toc198039364)

[Краткое описание алгоритма объединения результатов: 32](#_Toc198039365)

[2.6 Реализация объединения результатов гибридной системы с результатами стохастической оптимизации 34](#_Toc198039366)

[2.7 Запуск и работа рекомендательной системы 35](#_Toc198039367)

[ЗАКЛЮЧЕНИЕ 38](#_Toc198039368)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 39](#_Toc198039369)

ПРИЛОЖЕНИЕ А Исходный код программы

# **ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ**

В настоящей выпускной квалификационной работе бакалавра применяют следующие термины с соответствующими определениями:

Коллаборативная фильтрация — ­это один из методов построения прогнозов (рекомендаций) в рекомендательных системах, использующий известные предпочтения (оценки) группы пользователей для прогнозирования неизвестных предпочтений другого пользователя

Стохастическая оптимизация — это метод оптимизации, используемый для решения задач, когда функционал, который нужно оптимизировать, не может быть вычислен точно или слишком сложен для аналитического решения

Датафрейм — это структура данных, используемая в библиотеке pandas для работы с табличными данными. Она представляет собой двумерную таблицу с метками строк и столбцов, где каждый столбец может содержать данные различных типов, например, числа, строки, булевы значения и т.д. DataFrame позволяет быстро и удобно обрабатывать и анализировать большие объемы данных, выполнять фильтрацию, сортировку, группировку, агрегацию, а также применять различные математические и статистические операции к данным

# **ВВЕДЕНИЕ**

Актуальность темы данной работы связана с тем, что в настоящее время туристические рекомендательные системы играют важную роль жизни общества и привлекают множество посетителей, которые хотят ознакомиться с экспонатами, памятниками, историей и культурой различных народов. Однако, туристы часто сталкиваются с проблемой ограниченного времени, что может вызвать у них чувство неудовлетворенности и привести к негативному опыту путешествия.

Для решения данной проблемы, туристические организации начали применять технологии и методы, которые могут помочь пользователям более эффективно планировать свое время и выбирать наиболее интересующие их места. В этом контексте, разработка рекомендательных систем становится все более актуальной и востребованной задачей, поскольку они могут помочь путешественникам выбирать наиболее важные места и оптимизировать свое время.

Однако, существующие рекомендательные системы не учитывают предпочтения посетителей и не предоставляют наиболее оптимальные рекомендации. В этой связи, разработка новых методов и алгоритмов рекомендательных систем с использованием современных методов стохастической оптимизации является важной задачей, которая может улучшить качество обслуживания посетителей и повысить удовлетворенность ими.

Разработка рекомендательной системы на базе решения задачи стохастической оптимизации для туристов имеет высокую актуальность в контексте улучшения качества обслуживания путешественников и оптимизации использования времени на посещение значимых мест.

Таким образом, выполненная работа актуальна и с научно-методической/теоретической, и с практической точек зрения.

Цель данной работы - разработать рекомендательную систему, использующую эвристику, оптимизирующую маршрут, которая будет предоставлять путешественникам персонализированные рекомендации на основе их предпочтений, интересов, а также учитывающая расположения объектов. Рекомендательная система будет использоваться для помощи туристам в выборе наиболее интересующих экспонатов, оптимизации использования времени на прохождение маршрута и повышения удовлетворенности посетителей.

Для достижения поставленной цели в работе были решены следующие задачи:

* изучение существующих методов и алгоритмов рекомендательных систем и выбор оптимальных для решения задачи;
* преобразование базы данных под рекомендательную систему;
* реализация нескольких типов рекомендательных систем;
* разработка объединения результатов работы нескольких алгоритмов рекомендательных систем;
* разработка компонентов рекомендательной системы, с помощью которых учитываются результаты решения задачи стохастического программирования по определению маршрута движения туриста с учетом различных сценариев;
* реализация и тестирование рекомендательной системы, результатом которой является близкий к оптимальному маршрут движения туриста.

Для выполнения работы были использованы различные инструменты и методы, включая:

1. Язык программирования Python. Python является одним из наиболее распространенных языков программирования, который широко используется для научных вычислений, машинного обучения и анализа данных. В данной работе Python использовался для разработки алгоритмов и реализации рекомендательной системы.
2. Файлы CSV: Формат CSV (Comma Separated Values) является одним из наиболее распространенных форматов хранения и обмена данными. В данной работе файлы CSV использовались для хранения и обработки данных, связанных с экспонатами, посетителями, результатов работы рекомендательных систем.
3. Библиотека SciPy: SciPy — это библиотека для научных вычислений в Python, которая содержит множество инструментов для решения задач оптимизации, линейной алгебры, статистического анализа и другого. В данной работе библиотека SciPy использовалась для реализации алгоритмов стохастической оптимизации.
4. Иные библиотеки Python для работы с данными. В данной работе использовались различные библиотеки Python для работы с данными, такие как библиотека os для работы с файловой системой, библиотека Pandas для анализа данных, библиотека NumPy для научных вычислений и так далее.

В рамках данной работы были получены следующие основные результаты:

1. Отсортированные рекомендации с баллами для двух алгоритмов рекомендательных систем:
   1. Коллаборативная фильтрация: для каждого туриста на основе его предыдущих посещений формируется список рекомендаций, который содержит объекты, наиболее похожие на те, которые посетитель уже посмотрел. После этого список рекомендаций сортируется по баллам, которые рассчитываются на основе сходства между объектами и предпочтений пользователя.
   2. Рекомендательная система, основанная на схожести объектов: для каждого турист на основе его предыдущих посещений формируется список рекомендаций, который содержит объекты, наиболее похожие на те, которые посетитель уже посмотрел. После этого список рекомендаций сортируется по баллам, которые рассчитываются на основе сходства между объектами.
2. Оптимизированный эвристикой маршрут для построения оптимального маршрута в заданный отрезок времени.
3. Отсортированный результат слияния рекомендательных систем по баллам. Для каждого туриста получены списки рекомендаций на основе двух различных алгоритмов рекомендательных систем. После этого списки рекомендаций объединены и отсортированы по баллам, полученным на основе результатов двух алгоритмов.
4. Итоговый отсортированный список рекомендаций: для каждого туриста формируется индивидуальный список рекомендаций, который учитывает результат слияния двух алгоритмов рекомендательных систем и работу эвристического алгоритма стохастической оптимизации. Список рекомендаций отсортирован по баллам, которые рассчитываются на основе сходства между объектами и предпочтений пользователя, а также результатов работы стохастической оптимизации. Кроме того, данные для каждого посетителя формируются индивидуально после посещения нового объекта, что позволяет рекомендовать объекты, наиболее соответствующие интересам и предпочтениям каждого конкретного туриста.

Все результаты были записаны в файлы CSV, которые могут быть использованы для последующего анализа и улучшения алгоритмов рекомендательной системы.

Результаты работы предназначены для внедрения в работу туристических агентств с целью улучшения качества обслуживания туристов и оптимизации времени, затрачиваемого на просмотр объектов. Разработанный алгоритм рекомендации может быть использован компаниями для предоставления индивидуальных рекомендаций пользователям, исходя из их предпочтений и интересов. Полученные результаты могут быть использованы для улучшения качества обслуживания и увеличения удовлетворенности пользователей, что, в свою очередь, может привести к увеличению посещаемости.

Кроме того, результаты работы могут быть использованы для дальнейших исследований в области рекомендательных систем и оптимизации процессов. Полученные результаты могут послужить основой для разработки более эффективных алгоритмов рекомендации и оптимизации процессов в туристических агентствах, что в свою очередь может привести к повышению качества обслуживания и удовлетворенности пользователей.

Также результаты работы могут быть использованы для интеграции с другими системами управления, такими как системы бронирования билетов, системы управления посетителями и системы анализа посещаемости. Это может помочь туристическим компаниям снизить затраты на управление и повысить эффективность работы.

В целом, использование разработанной рекомендательной системы позволяет туристическим агенствам улучшить качество обслуживания посетителей, повысить эффективность работы, уменьшить расходы на персонал и повысить прибыль.

# **1 ТЕОРИЯ**

## 1.1 Потребности в разработке гибридной рекомендательной системы на базе стохастической оптимизации для культурных учреждений

В современном мире наблюдается быстрый рост информации, что создает огромные проблемы для пользователей в выборе наиболее подходящей информации. Одним из решений этой проблемы являются рекомендательные системы, которые предоставляют пользователям персонализированные рекомендации на основе их предпочтений и интересов. Рекомендательные системы широко используются в различных областях, включая электронную коммерцию, социальные сети, видео- и музыкальные сервисы и т.д. Однако, разработка рекомендательных систем, которые бы удовлетворяли требованиям пользователей, остается актуальной и сложной задачей.

В сфере культуры и искусства рекомендательные системы также играют важную роль. Музеи, театры и другие культурные учреждения сталкиваются с проблемой обеспечения качественного обслуживания посетителей и увеличения их удовлетворенности. Рекомендательные системы могут помочь решить эту проблему, предоставляя посетителям персонализированные рекомендации посещения экспонатов или мероприятий на основе их интересов и предпочтений.

Однако, существующие рекомендательные системы не всегда удовлетворяют потребности пользователей. Например, некоторые системы не учитывают индивидуальные потребности и предпочтения пользователей, а другие могут оказаться неэффективными в силу недостаточной точности предсказания рекомендаций.

Именно поэтому разработка новых методов и алгоритмов для создания эффективных рекомендательных систем остается актуальной и важной задачей. В рамках данной работы рассматривается разработка гибридной рекомендательной системы на базе решения задачи стохастической оптимизации. Она позволит туристическим компаниям предоставлять посетителям персонализированные рекомендации на основе их интересов и предпочтений, что, в свою очередь, улучшит качество обслуживания и повысит удовлетворенность.

## 1.2 Анализ технических возможностей для реализации алгоритмов

Для работы алгоритмов достаточно слабопроизводительных устройств, таких как планшеты, которые обычно имеют ограниченные ресурсы. В связи с этим, снижено количество вычислений, которые производятся на устройствах, и используются максимально доступные данные для получения оценок пользователя. В данном случае мы используем время, проведенное пользователем у каждого объекта в качестве такой оценки.

Следует отметить, что в некоторых местах может быть ограничен доступ к интернету или локальной сети. Это означает, что все вычисления для рекомендательной системы должны иметь возможность выполняться в том числе на мобильных устройствах. Однако при этом необходимо обеспечивать при этом высокую точность рекомендаций.

## 1.3 Базы данных

База данных содержит информацию о посетителях, которые ранее использовали данную рекомендательную систему. Из этих данных извлекается информация о времени посещения каждого объекта каждым посетителем. Также в базе данных содержится информация о местах, включая их название, категорию, описание и номер в базе данных. Данная информация была использована для создания csv файлов с отобранными посетителями и экспонатами.

Кроме того, в базе данных содержится информация о расположении экспонатов в двухмерном пространстве. Из этих данных строится полный граф, в котором можно попасть в каждую точку из каждой точки.

## 1.4 Поиск существующих рекомендательных систем

Алгоритмы рекомендательных систем могут быть различными и, как правило, определяются конкретной задачей, которую необходимо решить. В литературе [1] существует множество алгоритмов, которые можно использовать для построения рекомендательных систем. Некоторые из них включают:

1. Content-based filtering (CBF) - основан на анализе характеристик продуктов или услуг, которые пользователь оценил или потребовал.
2. Collaborative filtering (CF) - основан на схожести пользователей, исходя из их предпочтений. Может быть двух типов: user-based и item-based.
3. Knowledge-based filtering - использует экспертную систему для генерации рекомендаций.
4. Hybrid recommender systems - комбинируют различные алгоритмы, такие как CBF и CF, чтобы улучшить качество рекомендаций.
5. Demographic-based filtering - использует информацию о демографических характеристиках пользователей, таких как возраст, пол, место жительства и т.д.
6. Knowledge-based filtering with demographic information - комбинирует информацию о демографии пользователей с экспертной системой для генерации рекомендаций.
7. Context-aware filtering - учитывает контекст, в котором пользователь просматривает или потребляет продукты, например, местоположение, время и т.д.
8. Social-based filtering - основан на анализе социальных связей пользователей и используется для генерации рекомендаций, основанных на социальных связях.

## 1.5 Сопоставление имеющихся данных и существующих алгоритмов

При разработке рекомендательных систем, одним из важных аспектов является учет демографических характеристик пользователей. Эти данные могут включать в себя возраст, пол, местоположение и т.д. Однако, сбор такой информации может стать проблемой, особенно в контексте музеев, где посетители могут заходить на короткий срок и не захотеть тратить время на заполнение анкет или опросников. Дополнительно, вопросы о демографических характеристиках могут вызвать недовольство и неудовольствие у пользователей, особенно если они считают, что эти вопросы не имеют никакого отношения к их посещению музея. Из-за этих факторов, сбор демографических данных может стать непрактичным и неэффективным для использования в рекомендательных системах.

В связи с этим, было решено использовать две системы, которые могут работать без дополнительных данных от пользователя: коллаборативную фильтрацию и рекомендательную систему, основанную на сходстве объектов.

Коллаборативная фильтрация основана на анализе предпочтений пользователей и сопоставлении их с предпочтениями других пользователей, чтобы предложить рекомендации.

Рекомендательная система, использующая сходство, основана на анализе характеристик объектов и сопоставлении их с предпочтениями пользователей.

Обе системы не требуют от пользователя дополнительных данных, поэтому возможно использование имеющиеся данные из базы данных.

Для использования коллаборативной фильтрации необходимо определить, какие пользователи могут быть схожими по предпочтениям и на основе этого определить, какие экспонаты можно порекомендовать. Для основанной на сходстве рекомендательной системы нам нужно определить характеристики объектов, которые будут использоваться для сопоставления с предпочтениями пользователей.

База данных музея содержит информацию о посетителях и экспонатах, которые могут быть использованы для выбора рекомендательных систем. Были собраны данные о времени, проведенном посетителями у каждого объекта. Эта информация была использована для определения, какие экспонаты могут быть наиболее интересны для посетителей. Также была получена информация о характеристиках объектов и их расположении в помещениях, что позволяет определить, какие экспонаты могут быть связаны друг с другом и могут быть интересны для пользователей, основываясь на их времени пребывания рядом с объектом.

Для учета расстояний между сущностей можно использовать данные о фактическом расположении экспонатов в музее, которые были извлечены из базы данных. Таким образом, можно более точно определить, какие объекты находятся ближе друг к другу, и использовать эту информацию для формирования рекомендаций.

## 1.6 Коллаборативная фильтрация

Коллаборативная фильтрация - одна из наиболее распространенных рекомендательных систем, используемых для предсказания рейтингов или предпочтений пользователей. Работа этой системы основана на предположении, что если два пользователя оценили некоторый набор объектов (фильмов, музыки, товаров и т.д.) схожим образом, то они будут иметь схожие вкусы и интересы [2].

В подходе, анализирующем пользователей, сначала вычисляется сходство между всеми парами пользователей, используя меры сходства, такие как косинусное сходство или корреляция Пирсона. Затем, для каждого пользователя находятся похожие на него пользователи, и вычисляются взвешенные оценки объектов на основе их оценок и сходства с выбранными объектами. Эти оценки затем используются для формирования рекомендаций.

Коллаборативная фильтрация имеет ряд преимуществ, таких как возможность учитывать неявные предпочтения пользователей.

Для реализации коллаборативной системы необходимо выполнить несколько шагов. Сначала нужно получить данные о предпочтениях пользователей, то есть оценки, которые они выставляют объектам. Затем эти данные нужно преобразовать в матрицу оценок. Обычно матрица оценок имеет вид таблицы, где каждая строка соответствует пользователю, каждый столбец – объекту, а ячейка таблицы содержит оценку, которую пользователь выставил объекту. Схематично данная рекомендательная система и матрица оценок представлены на рисунке 1.

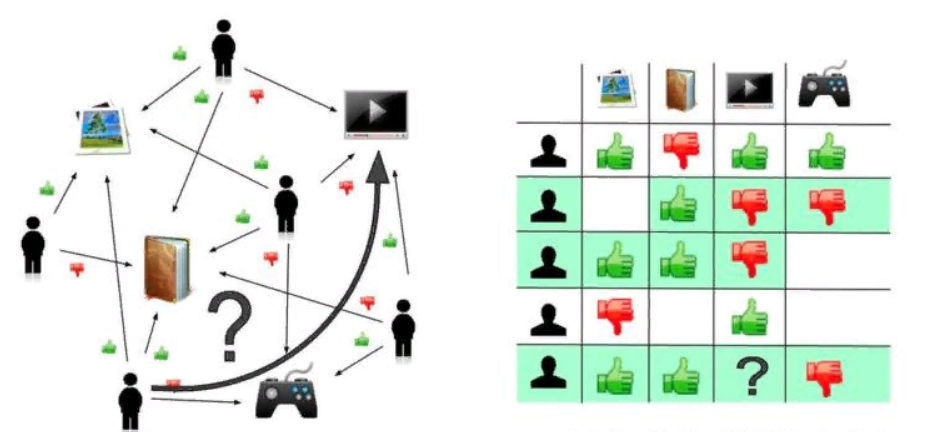


Рисунок 1 – Иллюстрация коллаборативной фильтрации

После этого можно перейти к этапу поиска похожих пользователей или объектов. Для этого используются различные алгоритмы, такие как метод k-ближайших соседей, сингулярное разложение и другие. Они позволяют определить наиболее похожие пользователей или объекты на основе сравнения их оценок.

Далее необходимо выполнить этап предсказания оценок. Для этого используются различные методы, такие как среднее значение оценок соседей или взвешенное среднее, где веса зависят от степени похожести соседей. На основе предсказанных оценок можно формировать список рекомендаций.

Важным моментом является оценка качества рекомендаций. Для этого используются различные метрики, такие как точность, полнота, F-мера и др. Также проводятся эксперименты на тестовых данных, чтобы оценить эффективность системы.

Одним из недостатков коллаборативной системы является холодный старт [3], то есть невозможность предложить рекомендации для новых пользователей или объектов. Также система может страдать от проблемы рекомендации известных объектов, то есть предлагать уже известные пользователю объекты.

Коллаборативная фильтрация эффективна, потому что она основана на анализе предпочтений пользователей и нахождении сходств между их предпочтениями. Это позволяет системе предлагать пользователю объекты, которые ему могут понравиться, на основе предпочтений их похожих пользователей.

Исследования показывают, что коллаборативная фильтрация является одной из самых эффективных и точных методов рекомендаций. Например, в статье [1] был проведен обзор и сравнение различных методов коллаборативной фильтрации, и результаты показали, что они превосходят другие методы рекомендаций по точности.

Также, исследование "Collaborative Filtering Recommender Systems" (2011) [4] выявило, что коллаборативная фильтрация имеет высокую точность и показывает хорошие результаты в рекомендациях в различных областях, включая музыку, фильмы, книги и т.д.

Из этих исследований следует, что коллаборативная фильтрация является эффективным методом рекомендаций, особенно в случае большого количества пользователей и объектов, что делает ее подходящим выбором для нашей задачи рекомендации экспонатов в музее.

## 1.7 Система на основе схожести объектов

Система рекомендаций на основе сходства объектов (item-based recommendation system) предполагает, что если пользователь понравился один объект, то ему могут понравиться и другие объекты, которые похожи на этот объект. Для построения такой системы сначала нужно создать матрицу сходства между объектами. Это можно сделать с помощью различных метрик сходства, таких как косинусное сходство или евклидово расстояние.

После того, как матрица сходства создана, для каждого объекта можно найти его похожие объекты. Например, если пользователь заинтересован в экспонате, который относится к разделу «История технологий», то ему могут быть интересны и другие экспонаты из этого же раздела. Для этого для каждого объекта находятся наиболее похожие на него объекты из того же раздела, после чего список объектов сортируется по убыванию степени сходства. Пользователю предлагаются наиболее похожие объекты из этого списка.

Основное преимущество системы рекомендаций на основе сходства объектов заключается в том, что она может предложить пользователю объекты, которые ему могут понравиться, но которые он сам бы не нашел. При этом не требуется большого количества информации о пользователе, достаточно знать, какой объект ему понравился, чтобы на основе сходства объектов подобрать другие, которые могут его заинтересовать.

Однако система рекомендаций на основе сходства объектов имеет и свои недостатки. В частности, она не учитывает индивидуальные предпочтения пользователя и его контекст, такой как время, место и настроение. Также может возникнуть проблема "холодного старта", когда новый объект, который еще не имеет достаточного количества оценок и связей с другими объектами, не будет учтен системой.

Для решения этих проблем, в систему можно добавить дополнительные факторы, такие как временные, географические или социальные данные. Это позволяет учитывать контекст и индивидуальные предпочтения пользователя, а также повышает качество рекомендаций.

Исследования показывают, что системы, основанные на сходстве объектов, могут быть эффективными в рекомендации объектов, которые могут заинтересовать пользователя. Например, исследование "Content-Based Recommender Systems: State of the Art and Trends" от 2014 года показало, что системы, основанные на сходстве объектов, могут быть эффективными в рекомендации музейных экспонатов, книг и фильмов.

Другое исследование "A Survey of Content-based Recommender Systems Based on Machine Learning Techniques" от 2016 года также подтвердило эффективность систем, основанных на сходстве объектов, и показало, что они могут быть эффективными в рекомендации объектов в различных областях, включая туристические маршруты по городам, музеям и галереям.

Таким образом, системы, основанные на сходстве объектов, могут быть эффективными в рекомендации объектов, которые могут заинтересовать пользователя. Данная рекомендательная система проиллюстрирована на рисунке 2.



Рисунок 2 – Content-based рекомендательная система

## 1.8 Постановка задачи метаэвристики

Рассмотрим набор узлов N = {1, ..., |N|}, где каждому узлу i ∈ N сопоставлен неотрицательный балл Si. Начальный и конечный узлы – 1 и |N| соответственно. Цель оптимизации – найти путь в пределах некоторого Tmax, посещающий подмножество N и максимизирующий общий собранный балл. Каждый узел можно посетить не более одного раза.

Задачу можно сформулировать как модель целочисленного программирования со следующими переменными: Xij = 1, если мы посетим узел i до узла j, и 0 в противном случае. tij – время, затрачиваемое на посещение узла

## 1.9 Метаэвристика HSATS

HSATS (Hybrid Simulated Annealing – Tabu Search) представляет собой метаэвристический подход, предназначенный для решения задач маршрутизации с ограничениями. Он сочетает в себе преимущества симулированного отжига (Simulated Annealing, SA) и поиска с запретами (Tabu Search, TS), чтобы повысить эффективность поиска глобального оптимума в условиях ограниченного пространства решений и стохастической природы данных.

Компоненты алгоритма:

1. Метод имитации отжига (SA). Используется для управления вероятностью приёма невыгодных (менее оптимальных) решений. Это позволяет избегать застревания в локальных максимумах. Температура постепенно снижается, уменьшая вероятность принятия плохих решений на поздних этапах.
2. Поиск с запретами (Tabu Search) Поддерживает список табу (tabu list), в который помещаются недавно рассмотренные решения. Это предотвращает циклический возврат к уже исследованным маршрутам и способствует более широкому охвату пространства решений.
3. Генерация соседей (Neighborhood strategies). На каждой итерации создаются соседние маршруты путём локальных модификаций:

* вставка новой точки;
* удаление точки;
* разворот подотрезка маршрута;
* комбинированная замена (вставка и удаление).

1. Целевая функция. Максимизируется сумма наград за посещение объектов. При этом учитываются ограничения на общее время маршрута (включая перемещения и обслуживание точек).

## 1.11 Сравнение HSATS с конкурентом

Исходя из анализа открытых наборов данных для этой задачи для тестирования алгоритма был разработан генератор данных, в котором координаты вершин генерировались случайным образом из нормального распределения с математическим ожиданием равным 0 и среднеквадратичным отклонением равным 20. На основе местоположений вершин высчитывалась матрица расстояний как сумма евклидова расстояния между каждой парой точек, умноженного на коэффициент 0.1, и значений, взятых случайным образом из равномерного распределения на интервале [0, 2]. Время обслуживания каждой точки равнялось сумме значения, взятого из равномерного распределения на интервале [0, 10], умноженного на коэффициент 0.1, и значения, взятого из равномерного распределения на интервале [0, 2]. Награды за посещения вершин брались случайным образом из нормального распределения с математическим ожиданием 25 и среднеквадратичным отклонением 5. Количество точек и временное ограничение регулировались вручную.

Для сравнения использовался решатель от OR-Tools типа «Dimension». Решатель маршрутов использует объект, называемый измерением, для отслеживания величин, которые накапливаются на маршруте транспортного средства, таких как время в пути или, если транспортное средство осуществляет погрузку и доставку, общий вес, который оно перевозит. В случае данной задачи в качестве веса использовалась награда за посещение вершины. Для решения был выбран один агент, для которого строился маршрут посещения вершин. Вместо ограничения расстояния было задано ограничение по времени маршрута. Поскольку OR-Tools работает только с целыми числами, а исходные сгенерированные данные представлены в виде float, то в соответствии с рекомендациями официальной документации, значения матрицы расстояний и временное ограничение были умножены на 10000, после чего они автоматически округлялись до целочисленных значений встроенными инструментами OR-Tools. Из-за данных преобразований длина любого маршрута увеличивалась в 10000 раз, но само решение не менялось. Величина округления мала и не оказывала существенного влияния на решение. Также в алгоритм были добавлены штрафы за непосещение локаций. Каждый пропущенный пункт назначения добавлял штраф к общему расстоянию. Затем решатель находил маршрут, который минимизировал общее расстояние и сумму всех штрафов за пропущенные местоположения.

В качестве второго алгоритма был реализован алгоритм HSATS [4]. Для генерации первоначального решения, доступного к оптимизации, использовался «жадный алгоритм». Необходимость «подбора» параметров имитационного отжига, используемого в данном алгоритме, усложняет работу и является существенным недостатком относительно алгоритма из OR-Tools. После генерации данных, интеграции решателя OR-Tools, реализации алгоритма HSATS была проведена серия экспериментов, результаты которых представлены в таблице 1.

Из результатов можно увидеть, что на малом количестве вершин HSATS может отрабатывать в десятки раз быстрее, чем OR-Tools, выдавая схожий результат целевой функции, однако разница довольно быстро нивелируется при увеличении количества вершин. Также увеличении количества вершин и увеличении длины маршрута, разница между значениями целевой функции алгоритмов HSATS и OR-Tools постепенно уменьшается, и они начинают выдавать схожие результаты.

Таблица 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **№** | **Алгоритм** | **N** | **TL** | **T** | **OV** | **MR** | **OV/MR (%)** |
| 1 | OR-Tools | 50 | 20 | 1.00263 | 304 | 1252 | 24.281 |
| HSATS | 0.05637 | 278 | 22.204 |
| Жадный алгоритм | 0.01845 | 234 | 18.690 |
| 2 | OR-Tools | 50 | 50 | 1.00271 | 696 | 1263 | 55.106 |
| HSATS | 0.19117 | 654 | 51.781 |
| Жадный алгоритм | 0.05236 | 615 | 48.693 |
| 3 | OR-Tools | 50 | 100 | 1.00269 | 1006 | 1208 | 83.278 |
| HSATS | 0.18015 | 975 | 80.712 |
| Жадный алгоритм | 0.061012 | 905 | 74.917 |

где N – общее количество узлов, TL – ограничение по времени, T – время работы алгоритма, OV – значение целевой функции, MR – сумма наград за посещение всех вершин.

## 1.13 Гибридная рекомендательная система

Каждая из рекомендательных систем имеет свои сильные и слабые стороны. Коллаборативная фильтрация работает на основе схожести предпочтений пользователей, а рекомендация на основе сходства объектов использует сходство между объектами для выдачи рекомендаций.

Одним из преимуществ объединения результатов рекомендательных систем является то, что это позволит снизить риск ошибок. Каждая из систем может допустить ошибку, что может привести к неверным рекомендациям. Однако объединение результатов позволит увеличить точность рекомендаций и снизить риск ошибок. Также это помогает нивелировать ошибки, вызванные недостаточным количеством данных или искажением результатов в одной из систем.

Кроме того, объединение результатов может увеличить разнообразие рекомендаций. Каждая из систем может выдавать рекомендации, которые схожи между собой. Однако, объединение результатов позволит увеличить разнообразие рекомендаций и предложить пользователю более широкий выбор.

Одним из главных плюсов такого подхода является уменьшение проблемы "холодного старта", то есть ситуации, когда рекомендательная система не может предложить рекомендации для нового посетителя, у которого еще нет истории посещений.

Объединение результатов также позволит учесть различные критерии рекомендаций, которые могут быть важны для разных пользователей. Например, один пользователь может предпочитать получать рекомендации на основе сходства объектов, а другой – на основе схожести предпочтений. Объединение результатов позволит учесть все эти критерии и выдать наиболее подходящие рекомендации для каждого пользователя.

Такая система может обеспечивать более точную рекомендацию для активных пользователей, так как учитывает их предпочтения, а также оценки, которые они давали объектам в прошлом.

В результате объединения в гибридную всех двух систем - коллаборативной фильтрации, системы на основе сходства объектов и системы на основе расстояний, мы получаем комплексную рекомендательную систему, которая учитывает разные аспекты предпочтений посетителей и повышает точность рекомендаций.

## 1.11 Слияние результатов гибридной системы с результатами стохастической оптимизации

Слияние результатов гибридной рекомендательной системы с эвристикой стохастической оптимизации является еще одним шагом в улучшении качества рекомендаций. Главным преимуществом использования стохастической оптимизации в качестве эвристики является то, что она может помочь в поиске оптимального решения, основываясь на статистических методах и случайных величинах.

Одним из главных вызовов при слиянии результатов гибридной системы с эвристикой является то, что стохастическая оптимизация не дает точных баллов для каждого объекта. Она работает с вероятностями и распределениями, что может быть не очень удобным для смешивания с другими рекомендательными системами, которые работают с точными оценками.

Чтобы решить эту проблему, мы можем использовать различные методы смешивания результатов, чтобы дать больший вес объектам, которые были лучше оценены в стохастической оптимизации. Мы можем также установить пороговое значение, при котором объекты с низкими оценками из стохастической оптимизации будут исключены из списка рекомендаций.

Важным моментом при слиянии результатов является также учет взаимодействия между пользователями и объектами. В случае стохастической оптимизации, каждый пользователь может получить свой собственный набор рекомендаций, основанный на его предпочтениях и интересах. При слиянии результатов, мы должны учитывать эти индивидуальные различия и попытаться дать каждому пользователю наиболее подходящие рекомендации.

# **2 ПРАКТИЧЕСКАЯ ЧАСТЬ**

**2.1 Используемый язык программирования**

Для создания рекомендательной системы был выбрал язык программирования Python, так как он является одним из наиболее подходящих языков программирования для решения данной задачи. Выбор языка программирования Python для данной работы обусловлен рядом его преимуществ, которые делают его наиболее подходящим инструментом для разработки рекомендательной системы на базе стохастической оптимизации.

Простота и интуитивность синтаксиса Python делает его доступным для использования в задачах программирования разного уровня сложности, что особенно важно в контексте данной работы.

Кросплатформенность Python позволяет использовать его на любой операционной системе, что облегчает работу с данными на различных устройствах. Так рекомендательная система не зависит от операционной системы планшета, на которой она будет запущена.

Наличие обширной коллекции библиотек и модулей Python, таких как NumPy, Pandas, Scikit-learn и os, делает его идеальным инструментом для работы с данными, машинного обучения и создания рекомендательных систем.

Поддержка функционального программирования Python делает код более простым и структурированным, что особенно важно для работы над сложными проектами.

Высокая скорость выполнения кода благодаря компиляции некоторых частей Python делает его подходящим выбором для задач, связанных с вычислительными процессами и обработкой больших объемов данных.

Интерактивная консоль Python позволяет быстро тестировать и отлаживать код, что ускоряет процесс разработки.

Простая и эффективная система управления пакетами Python, такая как pip, позволяет удобно устанавливать и управлять сторонними библиотеками, что также сильно упрощает разработку рекомендательной системы на базе стохастической оптимизации.

## 2.2 Используемые библиотеки и модули

Для успешной реализации рекомендательной системы были использованы различные библиотеки и инструменты. Краткое описание и обоснование использование библиотек приведено ниже.

Библиотека SciPy предоставляет множество функций и инструментов для научных и инженерных вычислений, включая построение матриц сходств. Одно из основных преимуществ SciPy - скорость работы. Она оптимизирована для вычислительных задач и предоставляет высокую производительность благодаря использованию языка программирования C. Более того, SciPy имеет богатую коллекцию функций для работы с различными типами данных и предоставляет мощный и гибкий функционал для настройки и управления вычислениями.

Библиотека os является стандартной библиотекой Python, которая предоставляет множество функций для работы с операционной системой, такие как чтение и запись файлов, управление директориями и многое другое. Одно из основных преимуществ os - его простота использования. Функции этой библиотеки имеют понятный синтаксис и предоставляют широкий спектр возможностей для работы с файловой системой. Кроме того, библиотека os позволяет писать переносимый код, который будет работать на разных операционных системах.

Библиотека pandas является одной из самых популярных библиотек для работы с данными в Python. Ее основное преимущество — это возможность работы с файлами csv и другими форматами данных. Pandas предоставляет мощный и гибкий функционал для чтения и записи данных, фильтрации, агрегации и многих других операций над ними. Одним из ключевых преимуществ pandas является скорость работы. Она оптимизирована для быстрой обработки больших объемов данных и предоставляет множество инструментов для ускорения вычислений. Кроме того, pandas предоставляет удобный и интуитивно понятный синтаксис, который делает ее доступной для использования даже новичкам в области анализа данных.

Библиотека numpy является одной из самых популярных библиотек для работы с массивами и матрицами в Python. Ее основное преимущество — это скорость работы и мощный функционал для работы с многомерными массивами данных. Numpy использует оптимизированный C-код для выполнения матричных операций, что позволяет ей работать очень быстро и обрабатывать большие объемы данных. Кроме того, numpy предоставляет множество инструментов для работы с матрицами, векторами, линейной алгеброй, численными методами и многими другими математическими операциями. Одним из дополнительных преимуществ numpy является ее способность работать с различными типами данных, включая числа с плавающей точкой, комплексные числа и другие. Кроме того, numpy имеет мощный и удобный интерфейс для интеграции с другими библиотеками, такими как pandas и matplotlib, что делает ее очень гибким и мощным инструментом для работы с данными и анализа данных.

## 2.3 Реализация коллаборативной фильтрации

В рамках данной ВКР была реализована коллаборативная фильтрация, которая позволяет предсказывать оценки пользователей на основе оценок, которые поставили похожие пользователи. Для реализации данного алгоритма были использованы следующие шаги:

1. Импортируются необходимые библиотеки: pandas, sklearn.metrics.pairwise и os.
2. Функция collaborative\_system принимает два аргумента: sources\_path и results\_path.
3. Далее, создаются две переменные - exhibit\_data\_path и current\_user\_path, которые содержат путь к файлам exhibit\_data.csv и current\_user.csv соответственно.
4. Проверяется наличие файлов exhibit\_data.csv и current\_user.csv по заданным путям при помощи модуля os. Если хотя бы один из файлов не найден, выбрасывается исключение типа FileNotFoundError.
5. Загружаются данные из файлов exhibit\_data.csv и current\_user.csv в виде таблиц pandas DataFrame с помощью метода pd.read\_csv.
6. Выполняются необходимые операции предобработки данных. Например, в коде выполняется объединение таблиц с помощью метода pd.concat и установление верхнего ограничения времени, потраченного на экспонат, при помощи метода clip.
7. Строится матрица взаимодействий пользователей с объектами, используя метод pivot\_table библиотеки pandas.
8. Вычисляется косинусное сходство между пользователями, используя метод cosine\_similarity из библиотеки sklearn.metrics.pairwise.
9. Применяется алгоритм коллаборативной фильтрации, который основывается на сходстве между пользователями и предсказывает оценку, которую пользователь бы дал объекту, на основе оценок, которые поставили похожие пользователи.
10. Результаты предсказаний сохраняются в CSV-файл recs\_collaborative.csv в заданную папку results\_path.

В качестве меры близости была выбрано косинусное сходство. Так, если мы рассматриваем пользователей как векторы, где каждый компонент представляет собой оценку пользователя для определенного элемента, то косинусное сходство может помочь определить, насколько похожи два пользователя на основе их оценок.

Косинусное сходство — это мера сходства между двумя векторами предгильбертового пространства, которая используется для измерения косинуса угла между ними.

Если даны два вектора признаков, A и B, то косинусное сходство, cos(θ), может быть представлено используя скалярное произведение и норму (1)

, (1)

Также в работе реализована функция collaborative\_filtering\_validation, которая используется для проверки качества работы алгоритма коллаборативной фильтрации на подвыборке данных. Для проверки качества алгоритма используется средняя абсолютная ошибка между предсказанными и фактическими оценками.

Алгоритм валидации состоит из нескольких шагов. Сначала данные делятся на несколько частей, после чего алгоритм коллаборативной фильтрации применяется на каждой из частей с использованием подвыборки пользователей. Затем средняя абсолютная ошибка между предсказанными и фактическими оценками вычисляется для каждой из подвыборок. Если на всех подвыборках полученные результаты превышают предыдущие значения, то алгоритм считается успешно валидированным.

Для проверки качества работы алгоритма используются уже подгруженные из файла exhibit\_data.csv и current\_user.csv, которые предварительно загружаются в виде таблиц pandas DataFrame. После чего происходит предобработка данных, включающая объединение таблиц и установление верхнего ограничения времени, потраченного на экспонат.

Таким образом, валидация позволяет оценить точность работы алгоритма коллаборативной фильтрации на случайной подвыборке данных и сделать выводы о его эффективности и применимости к реальным данным.

По данным валидации на данных музея при выборке первой из пяти частей пользователей, отсортированных по схожести с искомым, погрешность предсказаний в секундах составляет около 15-20%.

## 2.4 Реализация рекомендательной системы, основанной на схожести объектов

При реализации рекомендательной системы, основанной на схожести объектов необходимо выбрать меру близости для векторов информации об объектах. В данной ВКР была использована косинусная мера сходства.

Косинусное сходство является одной из наиболее распространенных мер близости, используемых в рекомендательных системах и машинном обучении. Оно измеряет сходство между двумя векторами, определяя косинус угла между ними.

В контексте рекомендательных систем, косинусное сходство часто используется для измерения близости между пользователями или элементами. Например, если мы рассматриваем пользователей как векторы, где каждый компонент представляет собой оценку пользователя для определенного элемента, то косинусное сходство может помочь определить, насколько похожи два пользователя на основе их оценок. Аналогично, для элементов, косинусное сходство может помочь определить, насколько похожи два элемента на основе оценок, которые пользователи дали им.

Формула косинусного сходства проста для вычисления, быстро работает и легко интерпретируется. Она также не требует нормализации данных, что делает ее более удобной для использования в ситуациях, где данные могут иметь различный масштаб или единицы измерения. Все это делает косинусное сходство популярным выбором для реализации рекомендательных систем, основанных на контенте и коллаборативной фильтрации.

Краткое описание действий, производимых в коде:

* Импортируются необходимые библиотеки, включая pandas для работы с таблицами данных, sklearn для масштабирования данных и вычисления меры сходства, numpy для работы с массивами и матрицами данных, и os для работы с файловой системой.
* Функция content\_based используется для создания рекомендательной системы на основе контента, то есть для рекомендации объектов, похожих на последний посещенный пользователем объект музея.
* Задаются пути к файлам с данными пользователей и экспонатов в музее. Если указанные файлы не найдены, выбрасывается исключение FileNotFoundError.
* Загружаются данные о текущем пользователе и описания экспонатов музея в виде таблиц pandas DataFrame.
* Определяется ID последнего посещенного пользователем объекта музея.
* Из таблицы экспонатов выбираются релевантные признаки для вычисления меры сходства.
* Категориальные признаки закодированы как числовые, используя функцию pd.get\_dummies.
* Вычисляется косинусное сходство между всеми парами экспонатов в музее.
* Находятся наиболее похожие на последний посещенный пользователем объекты музея с помощью функции \_recommend\_wines. Для этого происходит сортировка объектов музея по косинусному сходству с последним посещенным пользователем объектом.
* Проверяется качество работы алгоритма рекомендации контента с помощью валидации. Для этого происходит разделение выборки на пять частей, вычисление средней косинусной меры сходства между каждой частью и последним посещенным пользователем объектом, и проверка того, что эти меры сходства упорядочены по убыванию. Если валидация проходит успешно, результаты сохраняются в файле recs\_content\_based.csv. Если валидация не проходит, выбрасывается исключение ValueError.

Плюсы данного подхода:

1. Модульность: функция "content\_based" отвечает за рекомендательную систему, использующую контентную фильтрацию, и легко вызывается из других модулей.
2. Надежность: код включает проверки наличия файлов с данными и возвращает ошибку, если они не найдены, что помогает гарантировать, что программа не запустится с неполными данными.
3. Скорость: код использует библиотеки NumPy и Pandas для эффективной работы с данными. Он также использует функцию argsort из NumPy для быстрой сортировки по сходству.
4. Чистота: код четко разделяет данные и функциональность, что делает его легко читаемым и понятным для других разработчиков.
5. Проверка на валидность: код содержит функцию "validation\_result", которая проверяет, насколько хорошо рекомендательная система работает на определенных данных, и сообщает об ошибке, если результаты не соответствуют ожиданиям.
6. Использование научных библиотек: код использует научные библиотеки, такие как Scikit-learn, для упрощения реализации рекомендательной системы и повышения ее эффективности.

## 2.5 Реализация объединения данных всех рекомендательных систем

Для создания гибридной системы в работе использовался способ смешивания результатов <https://doi.org/10.1023/A:1021240730564>. Рейтинги всех двух систем нормализуются к баллам (от 0 до 100) и складываются. В работе предусмотрена установка коэффициентов влияния каждой рекомендательной системы на итоговый результат.

## Краткое описание алгоритма объединения результатов:

1. Импортируются необходимые библиотеки и модули: pandas и os.
2. Функция merge\_recommendations принимает пути к исходным файлам (sources\_path) и директории для сохранения результатов (results\_path), а также три коэффициента для определения весов для каждого из двух методов рекомендаций (content\_based\_coeff, collaborative\_coeff, distance\_based\_coeff).
3. Функция загружает три файла рекомендаций: recs\_collaborative.csv, recs\_content\_based.csv
4. Если файл recs\_content\_based.csv не существует, возникает ошибка FileNotFoundError.
5. Функция вычисляет "оценки" для каждого из методов рекомендаций, используя максимальные значения сходства для метода на основе содержания.
6. Если файл recs\_collaborative.csv существует, функция вычисляет также оценки для метода на основе коллаборативной фильтрации и суммирует оценки для каждого объекта в одно общее значение "score".
7. Функция объединяет две таблицы по ID объекта, суммирует "оценки" для каждого метода и сохраняет их в файле merged\_recommendations.csv.
8. Если файл recs\_collaborative.csv не существует, функция создает пустой датафрейм, добавляет отсутствующие ID из файлов recs\_content\_based.csv и recs\_distances.csv со значением "score" равным 0, исключает из него объекты, которые уже посетил пользователь, и затем объединяет все три таблицы.

Данная реализация позволяет объединять рекомендации, полученные с помощью двух разных алгоритмов: коллаборативной фильтрации, контентной фильтрации и алгоритма на основе расстояний. Такое объединение позволяет получать более точные и разнообразные рекомендации, что улучшает качество сервиса и повышает удовлетворенность пользователей.

Преимущества данного кода также включают возможность гибкой настройки весовых коэффициентов для каждого алгоритма, что позволяет более точно настроить соотношение между разными типами рекомендаций и настроить сервис под конкретные потребности пользователей. Кроме того, код также автоматически определяет отсутствующие рекомендации в каждом из двух файлов и заменяет их на нулевые значения, что обеспечивает полноту объединенных рекомендаций.

## 2.6 Реализация объединения результатов гибридной системы с результатами стохастической оптимизации

При объединении результатов гибридной системы с результатами стохастической оптимизации по аналогии с пунктом 2.5 выбрана нормализация к баллам. Однако возникает проблема, так как в результатах эвристического алгоритма изначально отсутствуют баллы.

Процесс распределения баллов для эвристической системы основан на идеи, что пользователь обычно предпочитает находиться вблизи своих предыдущих объектов исследования. Для этого система определяет количество строк в эвристическом файле и начисляет баллы, исходя из порядкового номера строки. Чем ближе строка к концу, тем больше очков она получит.

Полученные баллы затем нормализуются и преобразуются для использования вместе с баллами, полученными из других методов рекомендаций. Результаты каждого метода рекомендаций объединяются, и баллы для каждого объекта подсчитываются как максимальный балл из всех методов. Результаты сортируются по полученным баллам и сохраняются в файле для использования в качестве окончательных рекомендаций.

Описание работы алгоритма объединения результатов:

1. Импортируются необходимые модули: pandas и os.
2. Получаются пути к двум CSV-файлам: heuristic.csv и merged\_recommendations.csv, результаты которых будут использоваться для получения окончательного списка рекомендаций.
3. Проверяется, существуют ли оба файла по указанным путям. Если хотя бы один из них не найден, вызывается исключение.
4. Загружаются данные из файлов в соответствующие объекты DataFrame: heuristic и recommendations\_df.
5. Из эвристических результатов оставляются только объекты, присутствующие в списке объединенных рекомендаций.
6. Вычисляется максимальное количество очков, которое можно получить из списка объединенных рекомендаций.
7. Для каждого объекта в списке эвристических результатов вычисляется количество очков на основе его порядкового номера в списке. Этот показатель преобразуется таким образом, чтобы максимальное значение было равно максимальному количеству очков из списка объединенных рекомендаций, а затем умножается на коэффициент эвристики.
8. Объединяются два DataFrame - эвристический и объединенный список рекомендаций - по ID объектов.
9. Для каждого объекта вычисляется максимальное количество очков, которое он может получить из объединенного списка рекомендаций и из эвристического списка.
10. Создается новый столбец 'Optimal\_Scores' в DataFrame, который содержит максимальное количество очков для каждого объекта.
11. Удаляются столбцы 'merge\_scores' и 'heuristic\_scores'.
12. DataFrame сортируется по столбцу 'Optimal\_Scores' в порядке убывания.
13. Полученный DataFrame сохраняется в файл final\_results.csv.
14. Выводится сообщение об успешном выполнении задачи.

## 2.7 Запуск и работа рекомендательной системы

Запуск рекомендательной системы происходит через файл main.py. При работе программы используется распараллеливание процессов для ускорения работы программы.

Процесс работы программы:

1. Выводится сообщение о запуске программы.
2. Считывается файл конфигурации config.yaml, который содержит пути к файлам данных и коэффициенты для вычисления баллов рекомендаций.
3. Используется модуль Julia для выполнения вычислений на Julia в файле heuristic.jl.
4. Запускаются функции, которые соответствуют системам рекомендаций: content\_based, distance, collaborative\_system. Они запускаются в отдельных процессах.
5. После того, как процессы систем рекомендаций завершены, вызываются функции merge\_recommendations и interference\_to\_euristic.
6. Функция merge\_recommendations объединяет рекомендации из всех двух систем и вычисляет окончательные баллы для каждой рекомендации.
7. Функция interference\_to\_euristic добавляет баллы на основе эвристической системы, которая оценивает экспоненциальную функцию, вычисленную на основе позиции объекта в списке и количества общих объектов с последним просмотренным объектом пользователем.
8. Выводится сообщение об успешном завершении программы и время, затраченное на ее выполнение.

Пример вывода при корректном выполнении программы представлен на рисунке 3.

Пример использования программы в реальной жизни представлен на рисунке 4.

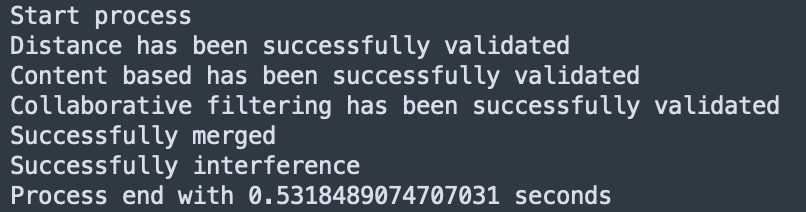
****

Рисунок 3 - Вывод окна терминала

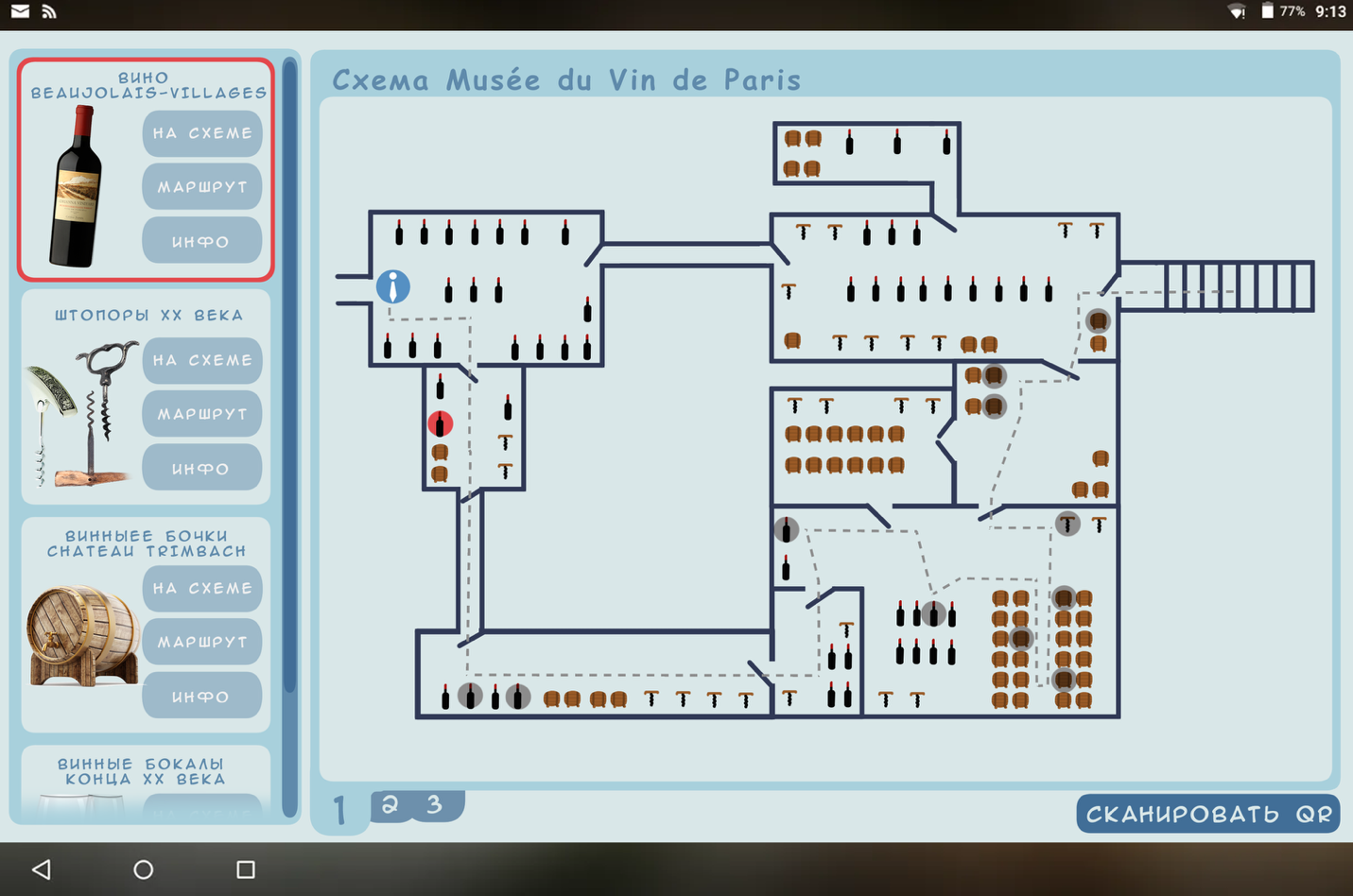


Рисунок 4 - Пример готового маршрута

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения выпускной квалификационной работы магистра было создано несколько рекомендательных систем для музея с использованием различных подходов: на основе сходства объектов, коллаборативной и на основе расстояния. Был реализован и сравнен с конкурентами алгоритм оптимизации HSATS. В результате были получены списки рекомендаций для пользователей, основанные на их предпочтениях, а также на основе расположения объектов в пространстве.

Оценка полноты решений показала, что системы достаточно хорошо справляются с задачей рекомендации объектов на основе различных критериев. В том числе, результаты валидации для каждой из систем показали высокую точность и эффективность.

Рекомендуется использовать полученные результаты для улучшения пользовательского опыта. Также можно использовать полученные данные для анализа поведения пользователей и оптимизации размещения экспонатов в музее.

Оценка технико-экономической эффективности показали, что внедрение рекомендательных систем может улучшить качество обслуживания и удовлетворенность пользователей, что может привести к увеличению доходов туристических компаний.

Научно-технический уровень выполненной работы находится на уровне лучших достижений в этой области. Использование различных методов и подходов позволило повысить научно-технический уровень данной работы и создать эффективные системы рекомендаций для туристов, что может быть полезно в других областях, где необходимо рекомендовать объекты на основе пользовательских предпочтений.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Burke, R. Hybrid Recommender Systems: Survey and Experiments / R. Burke. - User Modeling and User-Adapted Interaction, 2002. - Vol. 12, No. 4. - P. 331-370. - URL: <https://doi.org/10.1023/A:1021240730564> (дата обращения: 20.04.2023). - Текст: электронный.
2. Su, X. A Survey of Collaborative Filtering Techniques / X. Su, T. M. Khoshgoftaar. - Advances in Artificial Intelligence, 2009. - P. 421425. - URL: <https://downloads.hindawi.com/archive/2009/421425.pdf> (дата обращения: 20.04.2023). - Текст: электронный.
3. Bobadilla, J. A collaborative filtering approach to mitigate the new user cold start problem / J. Bobadilla, F. Ortega, A. Hernando, J. Bernal. - Knowledge-Based Systems, 2012. - Vol. 26. - P. 225-238. - URL: <https://www.sciencedirect.com/science/article/abs/pii/S0950705111001882> (дата обращения: 20.04.2023). - Текст: электронный.
4. Ekstrand, M. D. Collaborative Filtering Recommender Systems / M. D. Ekstrand, J. T. Riedl, J. A. Konstan. - Foundations and Trends® in Human–Computer Interaction, 2018. - Vol. 11, No. 3–4. - P. 113-239. - URL: <https://ieeexplore.ieee.org/document/8187128> (дата обращения: 20.04.2023). - Текст: электронный.

**ПРИЛОЖЕНИЕ А**

**Исходный код программы**

QR-код репозитория с кодом представлен на рисунке А.1.

МЕСТО ДЛЯ КОДА

Рисунок А.1 — QR-код репозитория