РЕФЕРАТ

Выпускная квалификационная работа магистра состоит из 44 страниц, 5 рисунков, 1 таблицы, 12 использованных источников, 1 приложения.

РЕКОМЕНДАТЕЛЬНАЯ СИСТЕМА, КОЛЛАБОРАТИВНАЯ ФИЛЬТРАЦИЯ, СХОДСТВО ОБЪЕКТОВ, ЭВРИСТИКА, МЕТАЭВРИСТИКА, ОПТИМИЗАЦИЯ, МАРШРУТ, ГИБРИДНАЯ СИСТЕМА, АРХИТЕКТУРА

Объектом разработки является гибридная рекомендательная система для туристов. Цель работы — создание системы, предоставляющей персонализированные рекомендации и оптимизирующей маршрут с учётом предпочтений и временных ограничений пользователя. В работе использованы методы коллаборативной фильтрации, анализа сходства объектов и метаэвристика HSATS, основанная на имитации отжига и табу-поиске. Разработан программный прототип, обеспечивающий высокую точность рекомендаций и формирование маршрутов с максимальной полезностью. Система может применяться в музеях, туристических агентствах и мобильных приложениях. Внедрение системы способно повысить удовлетворённость пользователей и экономическую эффективность туристических услуг. Прогнозируется расширение функционала системы и её интеграция с внешними сервисами.

СОДЕРЖАНИЕ

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящей выпускной квалификационной работе магистра 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

Эвристика — алгоритм решения задачи, включающий практический метод, не являющийся гарантированно точным или оптимальным, но достаточный для решения поставленной задачи.

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:

1) изучение существующих методов и алгоритмов рекомендательных систем и выбор оптимальных для решения задачи;
2) преобразование базы данных под рекомендательную систему;
3) реализация нескольких типов рекомендательных систем;
4) разработка объединения результатов работы нескольких алгоритмов рекомендательных систем;
5) разработка компонентов рекомендательной системы, с помощью которых учитываются результаты решения задачи стохастического программирования по определению маршрута движения туриста с учетом различных сценариев;
6) 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:
    a. 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.
    b. 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 агентствам улучшить качество обслуживания посетителей, повысить эффективность работы, уменьшить расходы на персонал и повысить прибыль.

1 ТЕОРИЯ

1.1 Потребности в разработке гибридной рекомендательной системы

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

Данные содержат информацию о посетителях, которые ранее использовали данную рекомендательную систему. Из этих данных извлекается информация о времени посещения каждого объекта каждым посетителем. Также данные содержат информацию 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 существует множество алгоритмов, которые можно использовать для построения рекомендательных систем [1]. 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 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 им может посоветовать рекомендательная система. Для основанной на сходстве рекомендательной системы нам нужно определить характеристики объектов, которые будут использоваться для сопоставления с предпочтениями пользователей.

1.6 Коллаборативная фильтрация

Коллаборативная фильтрация (КФ) по праву считается одним из столпов современных рекомендательных систем, обеспечивая мощный механизм для прогнозирования пользовательских интересов. В основе этого подхода лежит интуитивно понятное, но весьма эффективное предположение: люди, которые в прошлом демонстрировали схожие предпочтения или поведение, с высокой вероятностью будут иметь аналогичные интересы в будущем. Главная цель КФ — идентифицировать группы пользователей, чьи вкусы максимально совпадают с вкусами целевого пользователя, и затем использовать эту информацию для генерации персонализированных рекомендаций [2].

Суть работы коллаборативной фильтрации сводится к тщательному анализу исторических данных о взаимодействиях пользователей с различными объектами. Эти объекты могут быть чем угодно: от фильмов, книг и музыки до туристических достопримечательностей и экспонатов музея. Первым шагом в этом процессе является определение схожести между пользователями или между самими объектами. Для этого применяются различные математические метрики. Одной из наиболее широко используемых и эффективных мер схожести является косинусное сходство. Эта метрика вычисляет косинус угла между векторами предпочтений (или векторов характеристик объектов), что позволяет оценить их близость в многомерном пространстве. Чем ближе косинусное сходство к 1, тем больше схожесть.

После того как вычислены показатели схожести, коллаборативная фильтрация может функционировать в двух основных режимах:

Коллаборативная фильтрация на основе пользователей (User-Based Collaborative Filtering): В этом сценарии система активно ищет "единомышленников" — группу пользователей, чьи предпочтения тесно совпадают с предпочтениями текущего (целевого) пользователя. Как только эта группа идентифицирована, рекомендации формируются на основе действий и оценок, сделанных этими похожими пользователями. Например,

если группа пользователей, имеющих схожие интересы с туристом, регулярно посещает определенный тип выставок или высоко оценивает конкретные исторические памятники, то эти объекты будут предложены и целевому туристу. Это словно спросить у своих друзей, чьи вкусы вам хорошо знакомы, что бы они порекомендовали посмотреть.

Коллаборативная фильтрация на основе объектов (Item-Based Collaborative Filtering): здесь акцент смещается на схожесть между самими объектами. Система анализирует, какие объекты часто оцениваются или потребляются вместе, а также насколько похожи характеристики этих объектов. Если пользователь уже проявил интерес к одному объекту, система порекомендует ему другие объекты, которые статистически или семантически связаны с его уже предпочитаемыми. Например, если турист посетил музей современного искусства, система может предложить ему другую галерею, специализирующуюся на схожих направлениях, основываясь на том, что многие другие пользователи, которым понравился первый музей, также посещали и второй.

В контексте данной работы, была реализована коллаборативная фильтрация, ориентированная на пользователей, с использованием косинусного расстояния для определения схожести между векторами предпочтений пользователей. Это означает, что система активно ищет пользователей с "похожими вкусами" и использует их прошлый опыт для формирования рекомендаций.

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.

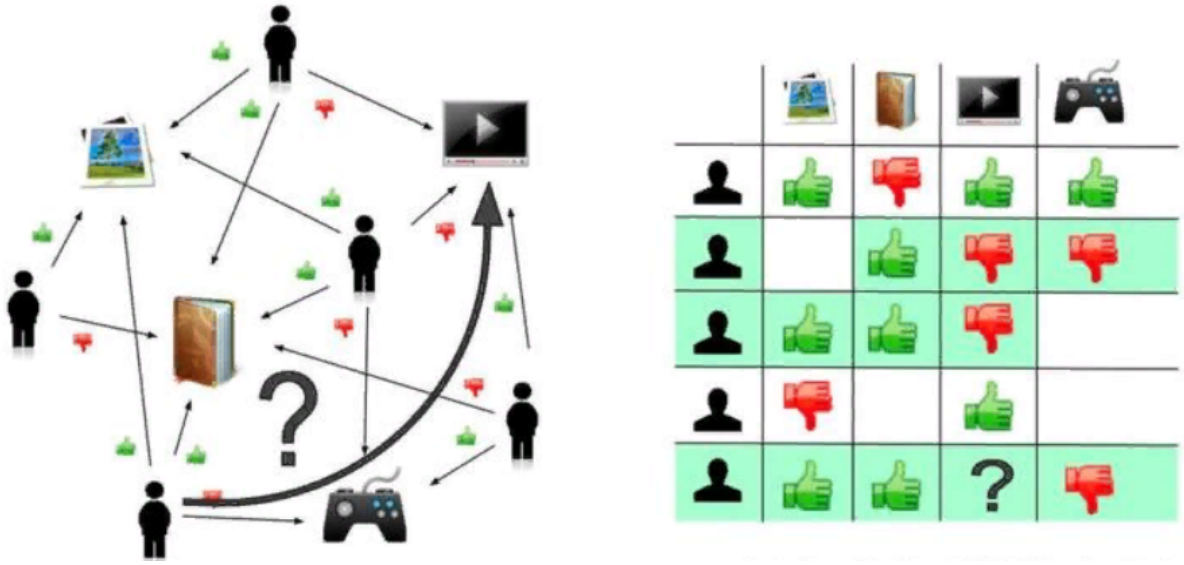

Рисунок 1 – Иллюстрация коллаборативной фильтрации

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, то есть невозможность предложить рекомендации для новых пользователей или объектов [3]. 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 был проведен обзор и сравнение различных методов рекомендаций, и результаты показали, что они коллаборативная фильтрация превосходит другие методы рекомендаций по точности [4].

Также, дополнительное исследование выявило, что коллаборативная фильтрация имеет высокую точность и показывает хорошие результаты в рекомендациях в различных областях, включая музыку, фильмы, книги и т.д [5].

Из этого 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, одно из исследований показало, что системы, основанные на сходстве объектов, могут быть эффективными в рекомендации музейных экспонатов, книг и фильмов [6].

Другое исследование 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 [7].

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 $i \in N$ сопоставлен неотрицательный балл S_i. Начальный и конечный узлы – 1 и $|N|$ соответственно. Цель оптимизации – найти путь в пределах некоторого T_{max},

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: $X_{ij} = 1$, если мы посетим узел i до узла j, и 0 в противном случае. t_{ij} – время, затрачиваемое на посещение узла.

1.9 Метаэвристика HSATS

HSATS (Hybrid Simulated Annealing–Tabu Search) — это гибридный алгоритм, который решает стохастическую задачу ориентирования. Он предназначен для построения априорного маршрута, то есть маршрута, спланированного до начала выполнения заданий автономным агентом, когда известны только вероятностные характеристики времени в пути и времени обслуживания, но не их конкретные значения.

HSATS [8] объединяет в себе два подхода: имитацию отжига [9] и табу-поиск [10]. Имитация отжига используется для принятия менее выгодных решений с определённой вероятностью, что помогает алгоритму избегать локальных минимумов. Табу-поиск, в свою очередь, отслеживает уже рассмотренные решения и не допускает возврат к ним, что предотвращает зацикливание. В каждой итерации алгоритм генерирует несколько соседних решений с помощью четырёх типов локальных модификаций маршрута: вставка новой точки, удаление уже включённой, реверс порядка посещения части маршрута и комбинированная вставка с удалением. Из всех кандидатов выбирается наилучший с учётом запретов табу-листа, и затем применяется критерий Метрополиса для решения, стоит ли его принять [11]. На каждом шаге температура в алгоритме отжига понижается, тем самым уменьшая вероятность принятия ухудшающих решений с течением времени.

Главное достоинство HSATS заключается в его способности эффективно исследовать пространство решений и находить качественные маршруты даже в условиях высокой неопределённости [12]. Он показывает

высокую точность: при тестировании на стандартных задачах он достигал результатов, сопоставимых или лучших, чем классические методы вроде ветвей и границ. Особенно хорошо алгоритм проявил себя на задачах большого масштаба, где точные методы становятся неприемлемыми из-за времени вычислений. Благодаря использованию стохастического компонента и гибкой системы локальных изменений, HSATS способен адаптироваться к разнообразным условиям задачи и учитывать риск несвоевременного возвращения.

Основной недостаток HSATS — это вычислительная тяжеловесность. Алгоритм требует большого количества итераций, и на каждом шаге необходимо вычислять вероятности соблюдения временного бюджета, что особенно затратно при наличии стохастических входных данных. Кроме того, сложность реализации выше, чем у классических жадных или локальных методов, так как требуется тщательно настраивать параметры охлаждения, длину табу-листа и систему оценки допустимости решений. В реальном времени HSATS не применяется напрямую: он слишком медленный для немедленной адаптации маршрута на ходу, поэтому он используется только как базовый этап — построение априорного маршрута перед началом движения.

1.10 Работа алгоритма

Сначала задаются параметры: начальная температура, минимальная температура, скорость охлаждения, длина табу-списка и число соседей, создаваемых на каждом шаге. Формируется начальное решение, которое считается текущим и одновременно лучшим на данный момент.

На каждом этапе создаётся несколько новых маршрутов, каждый из которых получается путём небольшого изменения текущего: вставки новой точки, удаления уже включённой, перестановки порядка или замены одной точки на другую. Эти маршруты называются соседями. Если какой-либо из них находится в табу-списке, он пропускается, за исключением случая, когда

его качество превышает качество лучшего найденного маршрута — тогда запрет игнорируется.

Среди оставшихся соседей выбирается наиболее перспективный. Если он лучше текущего маршрута, он принимается. Если хуже — может быть принят с определённой вероятностью, зависящей от текущей температуры. Такая стратегия позволяет выходить из локальных экстремумов.

После принятия решения маршрут обновляется. Если он оказался лучше найденных ранее, он сохраняется как новый лучший. Табу-список обновляется, чтобы временно запретить возврат к недавно пройденным решениям.

Когда заданное число шагов при текущей температуре выполнено, температура понижается. Процесс продолжается до тех пор, пока температура не опустится ниже порогового значения.

В результате возвращается лучшее найденное решение — маршрут, максимально соответствующий целевой функции при соблюдении всех ограничений.

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. 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 – Сравнение эффективности алгоритмов

№	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.

1.12 Гибридная рекомендательная система

Современные рекомендательные системы представляют собой мощный инструмент персонализации, призванный повысить удобство пользователя и эффективность взаимодействия с информационным пространством. Однако в реальной практике практически ни одна из существующих моделей не является универсальной и безупречной. Каждая из них обладает как сильными, так и слабыми сторонами, и это делает особенно актуальным вопрос о возможности интеграции различных подходов для достижения более устойчивых и точных результатов.

В частности, коллаборативная фильтрация и рекомендации на основе сходства объектов представляют собой два наиболее распространённых и широко используемых подхода в системах рекомендаций. Коллаборативная фильтрация работает, исходя из предположения, что пользователи с похожими интересами в прошлом будут делать схожие выборы в будущем. Такой подход особенно хорошо работает при наличии большого объёма пользовательских данных, поскольку на основе этих данных можно построить детализированную модель предпочтений. Однако, если история взаимодействия пользователя с системой скудна или вовсе отсутствует, эффективность этого метода резко снижается. Это особенно ярко проявляется в проблеме так называемого «холодного старта», когда система не способна выдать адекватные рекомендации для новых пользователей.

В противоположность этому, модели на основе сходства объектов опираются не на поведение пользователей, а на характеристики самих объектов. Они анализируют, насколько объекты, ранее интересовавшие пользователя, похожи на другие доступные элементы, и на основе этой информации формируют список рекомендаций. Такие системы менее чувствительны к отсутствию пользовательской истории, что делает их более универсальными в ряде случаев. Однако и они имеют свои ограничения: например, при слишком однотипных характеристиках объектов рекомендации могут стать однообразными и неинтересными.

В свете вышеизложенного, особое значение приобретает комбинированный подход, заключающийся в объединении результатов нескольких рекомендательных систем. Такой подход позволяет не только нивелировать недостатки каждой отдельной модели, но и создать более сбалансированный и устойчивый механизм рекомендаций. Одним из наиболее ощутимых преимуществ комбинирования является повышение точности: если одна модель по каким-либо причинам ошибается или работает нестабильно (например, из-за искажения данных или временного отсутствия информации), другая может компенсировать этот пробел, тем самым улучшая общее качество вывода.

Кроме того, объединение разных методов способствует уменьшению риска ошибок, которые могут возникнуть при использовании только одной модели. Каждый алгоритм имеет свою внутреннюю логику и особенности работы, а значит, способен по-своему интерпретировать поведение пользователя. Объединяя результаты, можно получить усреднённую, но более надёжную рекомендацию, которая будет учитывать разнообразные аспекты пользовательских предпочтений.

Ещё одно важное преимущество комбинированного подхода — увеличение разнообразия рекомендаций. Если каждый из алгоритмов склонен выдавать схожие предложения, то их объединение может привести к созданию более широкого и интересного набора вариантов. Это особенно ценно с точки зрения пользовательского опыта, поскольку помогает избежать повторяющихся или излишне похожих рекомендаций, делая взаимодействие с системой более живым и привлекательным.

Не стоит забывать и о том, что интеграция разных моделей в единую систему позволяет гибко настраивать приоритеты и веса отдельных компонентов. В зависимости от контекста применения, особенностей данных и поведения пользователей, можно адаптировать модель под конкретные условия. Это делает систему более адаптивной и способной эффективно

функционировать в различных сценариях — от рекомендаций культурных мероприятий до предложений контента в онлайн-сервисах.

Таким образом, использование комбинированного подхода в рекомендательных системах представляет собой рациональную стратегию повышения их эффективности и надёжности. За счёт объединения сильных сторон различных моделей достигается более высокая точность, увеличивается разнообразие рекомендаций, смягчается проблема холодного старта, а также обеспечивается устойчивая работа системы в условиях ограниченных или искажённых данных. Всё это делает комбинированные системы предпочтительным выбором при разработке практико-ориентированных решений в области интеллектуального анализа данных.

2 ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1 Используемый язык программирования

Для разработки рекомендательной системы на базе стохастической оптимизации был выбран язык программирования Python. Python считается одним из самых удобных и мощных инструментов в сфере анализа данных и интеллектуальных систем.

Во-первых, синтаксис Python интуитивно понятен, что значительно упрощает реализацию даже сложных алгоритмов. Это особенно важно в научных и прикладных проектах, где основное внимание уделяется логике решения, а не техническим тонкостям реализации.

Во-вторых, Python — кроссплатформенный язык, а значит, разработанная система может быть запущена на любой операционной системе — от настольных ПК до мобильных и встраиваемых устройств. Это делает её удобной в эксплуатации, особенно в условиях, где предполагается использование, например, на планшетах или других автономных устройствах.

Ещё одним весомым аргументом является богатая экосистема библиотек, охватывающая всё: от численных расчётов (NumPy) и работы с таблицами (Pandas), до машинного обучения (Scikit-learn) и файловых операций (os). Это позволяет сосредоточиться на логике рекомендательной системы, используя готовые и проверенные инструменты.

Python также поддерживает функциональный стиль программирования, что позволяет писать лаконичный и структурированный код. Это упрощает поддержку и масштабирование проекта.

Несмотря на то, что Python является интерпретируемым языком, многие его библиотеки используют низкоуровневые оптимизации, обеспечивая высокую производительность при обработке больших объёмов данных.

Наконец, наличие интерактивной среды разработки (например, Jupyter Notebook) и удобного менеджера пакетов (pip) делает Python особенно

удобным для прототипирования, тестирования и быстрой итеративной разработки.

Благодаря всем этим особенностям, Python стал логичным и эффективным выбором для реализации рекомендательной системы, опирающейся на методы стохастической оптимизации.

Исходный код разработки представлен в приложении А.

2.2 Используемые библиотеки и модули

Для реализации рекомендательной системы были задействованы ключевые библиотеки Python, каждая из которых сыграла свою роль в обеспечении эффективности и удобства разработки. Ниже представлены их краткие характеристики и обоснование выбора.

Библиотека SciPy применяется для выполнения научных и инженерных вычислений, в частности — для построения матриц сходства между объектами. Её основное преимущество — высокая производительность, достигнутая за счёт использования низкоуровневого кода на C. SciPy предоставляет обширный набор функций для линейной алгебры, оптимизации и статистики, что делает её мощным инструментом при решении задач численного анализа.

Модуль os — стандартная часть Python, используемая для взаимодействия с операционной системой. Он позволяет удобно работать с файловой системой: создавать, удалять и перемещать файлы и директории, а также считывать информацию о структуре каталогов. Простота синтаксиса и кроссплатформенность делают os незаменимым инструментом для написания переносимого кода.

Библиотека pandas является де-факто стандартом для работы с табличными данными в Python. Она обеспечивает гибкое и быстрое управление данными, включая чтение CSV-файлов, фильтрацию, агрегацию и трансформации. Благодаря удобному интерфейсу и оптимизации под

обработку больших объёмов данных, pandas стала важным инструментом для анализа и предобработки информации в рекомендательной системе.

NumPy — основа числовых вычислений в Python. Она предоставляет мощные средства для работы с многомерными массивами и поддерживает широкие возможности линейной алгебры, статистики и матричных операций. Высокая скорость работы достигается за счёт использования низкоуровневых реализаций. NumPy также легко интегрируется с другими библиотеками (например, pandas и matplotlib), что делает её неотъемлемой частью научного и прикладного программирования.

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) 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 между объектами было выбрано косинусное сходство. Данный подход позволяет эффективно оценивать степень схожести между пользователями, рассматриваемыми как векторы, компоненты которых отражают оценки, выставленные тем или иным элементам. Косинусное сходство измеряет угол между двумя такими векторами, что позволяет определить, насколько близки предпочтения пользователей друг к другу, независимо от масштаба их оценок. Это делает данный метод особенно подходящим для рекомендательных систем, где важно учитывать направление предпочтений, а не их абсолютные значения.

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)

$$\text{similarity} = \cos(\theta) = \frac{\mathrm{A}\cdot\mathrm{B}}{||\mathrm{A}||\cdot||\mathrm{B}||}, \quad (1)$$

В рамках работы была реализована функция collaborative_filtering_validation, предназначенная для оценки

качества алгоритма коллаборативной фильтрации на основе случайной подвыборки данных. В качестве метрики точности используется средняя абсолютная ошибка (MAE) между предсказанными и фактическими оценками.

Процесс валидации включает несколько последовательных этапов. Сначала исходные данные разбиваются на несколько подвыборок. Затем на каждой из них применяется алгоритм коллаборативной фильтрации с использованием ограниченного набора пользователей. Для каждой подвыборки рассчитывается средняя абсолютная ошибка. Если на всех этапах новые значения MAE оказываются ниже или сопоставимы с предыдущими, алгоритм считается прошедшим проверку.

Для проведения валидации используются предварительно загруженные данные из файлов exhibit_data.csv и current_user.csv. Они считываются в формате pandas DataFrame, после чего проходит этап предобработки: таблицы объединяются, и вводятся дополнительные ограничения, такие как максимальное время, которое пользователь может провести у экспоната.

Такая проверка позволяет оценить устойчивость и точность работы алгоритма в условиях неполных данных, а также сделать обоснованные выводы о его эффективности и пригодности к использованию на реальных пользовательских данных.

2.4 Реализация рекомендательной системы 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:

1) 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;
2) функция content_based используется для создания рекомендательной системы на основе контента, то есть для рекомендации объектов, похожих на последний посещенный пользователем объект музея;
3) задаются пути к файлам с данными пользователей и экспонатов в музее. Если указанные файлы не найдены, выбрасывается исключение FileNotFoundError;
4) загружаются данные о текущем пользователе и описания экспонатов музея в виде таблиц pandas DataFrame.
5) определяется ID последнего посещенного пользователем объекта музея.
6) из таблицы экспонатов выбираются релевантные признаки для

вычисления меры сходства.

7) категориальные признаки закодированы как числовые, используя функцию pd.get_dummies;
8) вычисляется косинусное сходство между всеми парами экспонатов в музее;
9) 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;
10) 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 рекомендательных систем

Для создания гибридной системы в работе использовался способ смешивания результатов. 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);
3) функция загружает два 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 HSATS

Реализация алгоритма включает следующие модули:

1) модуль генерации данных (data.py):
    a. функция generate_normal_values формирует случайные координаты точек;

b. метод get_travel_times вычисляет матрицу времени перемещения между точками с использованием евклидовой метрики и добавлением случайного шума;
c. метод get_service_times генерирует время обслуживания в каждой точке;
d. метод generate_initial_route строит начальный маршрут, добавляя точки, пока общее время маршрута не превысит заданный лимит;

2) модуль генерации кандидатных решений (candidates_generator.py):
   a. класс CandidatesGenerator содержит методы модификации текущего решения: вставка случайной непосещённой точки в оптимальную позицию маршрута, случайное удаление точки из маршрута с проверкой на допустимость по временив реверс случайного подотрезка маршрута с проверкой допустимости, комбинация вставки новой точки и удаления одной из уже включённых;
   b. метод generate_candidate_solution случайным образом выбирает одну из стратегий для генерации соседнего решения;
   c. методы validate_solution и calculate_objective_time проверяют маршрут на соблюдение временного лимита и оценивают его длительность;
3) основной модуль алгоритма HSATS (heuristic.py):
   a. функция calculate_objective вычисляет целевую функцию — суммарное вознаграждение по маршруту;
   b. функция simulated_annealing реализует модифицированный алгоритм имитации отжига;
   c. функция get_solution запускает алгоритм с конкретными параметрами и сохраняет результат в CSV-файл.

2.7 Этапы работы HSATS

Алгоритм работы включает в себя следующие этапы:

1) инициализация. Методом generate_initial_route формируется допустимый начальный маршрут. Рассчитывается его оценка с помощью функции calculate_objective;
2) основной цикл отжига. Работает до достижения конечной температуры. Содержит вложенный цикл генерации и оценки новых маршрутов;
3) генерация кандидатов. Через класс CandidatesGenerator создаются соседние маршруты на основе текущего, при этом используется одна из четырёх стратегий мутации;
4) оценка и выбор. Все кандидаты сравниваются с текущим решением. Если найдено улучшение, оно принимается. В противном случае применяется вероятностное правило имитации отжига;
5) табу-стратегия. Для предотвращения зацикливания используется список запретных решений. Повторно предложенные кандидаты из табу-листа отклоняются;
6) понижение температуры. Температура уменьшается согласно коэффициенту охлаждения. Процесс продолжается до заданного минимального порога.

2.8 Архитектура рекомендательной системы

Архитектура системы построена модульно: на начальном этапе используется комбинация коллаборативной фильтрации и контентно-ориентированного анализа, что позволяет учесть как поведенческие данные пользователей, так и характеристики объектов. Полученные предварительные оценки объединяются и поступают в модуль оптимизации, где применяются эвристические методы, способные учитывать вероятностную природу данных и оптимизировать итоговый список рекомендаций с учётом пользовательских предпочтений и ограничений.

Этапы рекомендательной системы:

1) старт: на вход системе поступают исходные данные: профиль пользователя, история взаимодействий, а также описания объектов. Этот блок запускает параллельную обработку двумя основными подсистемами;
2) коллаборативная фильтрация: одна из ветвей обработки — коллаборативная фильтрация, где рекомендации формируются на основе сходства между пользователями или между объектами, с учётом их общих интересов. Здесь применяются методы, опирающиеся на поведенческие данные пользователей;
3) система на основе сходства признаков: параллельно работает вторая подсистема — контентно-ориентированный подход, анализирующий атрибуты объектов (например, жанр, теги, описание) и определяющий степень их релевантности конкретному пользователю. Этот модуль не зависит от других пользователей и ориентируется на индивидуальные предпочтения;
4) объединение: результаты двух подсистем поступают в модуль объединения, где формируется предварительный пул рекомендаций. Это может быть реализовано через взвешенное агрегирование оценок или более сложные методы, учитывающие уверенность каждой модели;
5) оптимизирующая метаэвристика: полученный список направляется в модуль стохастической оптимизации. Здесь применяется метаэвристика (HSATS), которая переупорядочивает список, стремясь максимизировать итоговую полезность рекомендаций с учётом вероятностных ограничений (например, ограниченного времени, изменяющихся интересов или оценки доверия к рекомендациям);
6) упорядоченный список объектов: на выходе формируется финальный упорядоченный список рекомендаций, адаптированный

под пользователя. Этот список уже прошёл как структурный анализ (гибридная фильтрация), так и фазу интеллектуального переранжирования с использованием эвристики.

Представленная на рисунке 3 схема отражает архитектуру конечной рекомендательной системы, основанной на гибридном подходе с последующим переранжированием результатов с помощью метаэвристики.

Рисунок 3 – Этапы рекомендательной системы

Пример использования программы в реальной жизни представлен на рисунке 4.

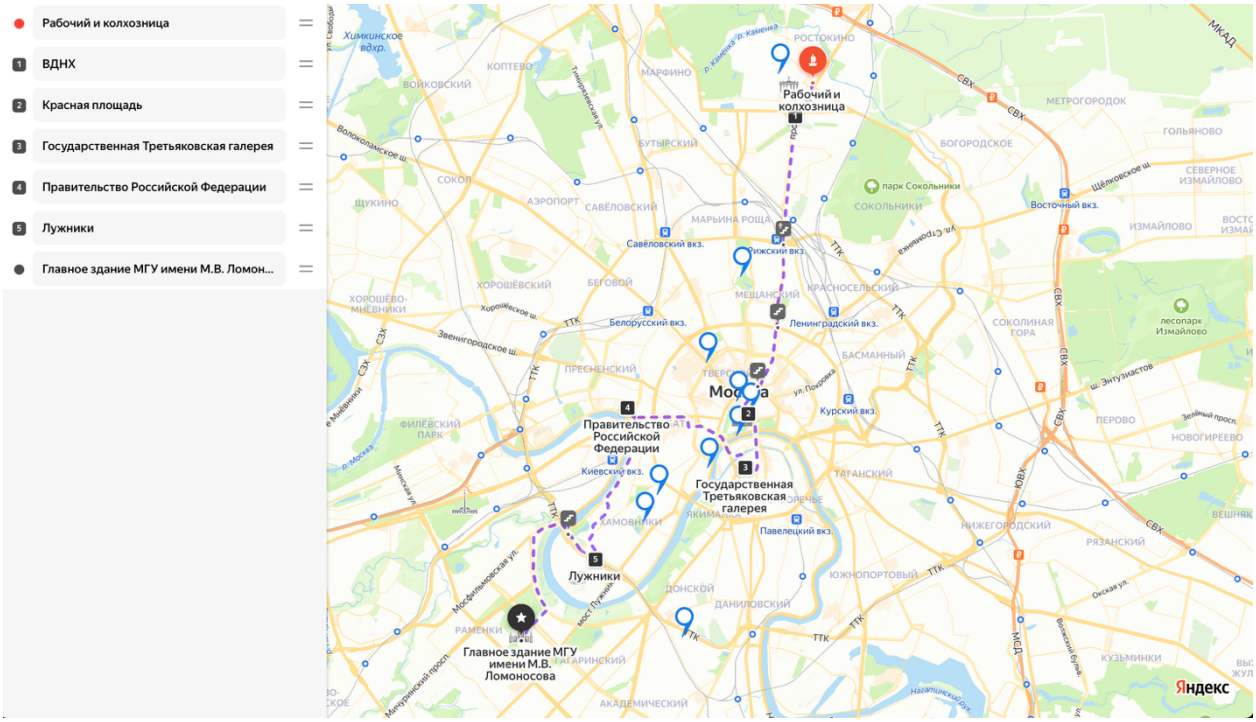

Рисунок 4 – Пример готового маршрута

ЗАКЛЮЧЕНИЕ

В результате проведённого исследования была разработана персонализированная рекомендательная система для туристов, основанная на применении методов стохастической оптимизации. Такой подход позволил учитывать как индивидуальные предпочтения пользователей, так и ограниченные ресурсы, включая время. Встраивание стохастической оптимизации в архитектуру системы повысило её адаптивность и обеспечило более точное формирование рекомендаций, ориентированных на конкретного пользователя. Разработанная система продемонстрировала эффективность гибридного подхода, сочетающего классические методы фильтрации и вероятностную эвристику. В дальнейшем планируется расширение функциональности за счёт внедрения дополнительных эвристических стратегий и совершенствование механизмов персонализации с целью повышения общей точности и устойчивости рекомендательного процесса.

Работа выполнена в полном объёме, её результаты подтверждены тестами и выражены в практической реализации.

Потенциальный положительный эффект от внедрения рекомендательных систем заключается в улучшении качества обслуживания и повышении удовлетворённости пользователей, что, в свою очередь, может способствовать росту доходов туристических компаний.

Научно-технический уровень выполненной работы соответствует современным требованиям и тенденциям в области разработки рекомендательных систем. Использование различных методов и подходов позволило повысить качество полученных решений и создать эффективные системы рекомендаций для туристов, что может быть полезно и в других областях, где требуется формирование рекомендаций 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: 23.05.2025).

2. Su X., Khoshgoftaar T. M. A Survey of Collaborative Filtering Techniques / X. Su, T. M. Khoshgoftaar. – Advances in Artificial Intelligence, 2009. – P. 421425. – URL: https://doi.org/10.1155/2009/421425 (дата обращения: 23.05.2025).

3. Bobadilla J., Ortega F., Hernando A., Bernal J. A collaborative filtering approach to mitigate the new user cold start problem / J. Bobadilla et al. – Knowledge-Based Systems, 2012. – Vol. 26. – P. 225–238. – URL: https://doi.org/10.1016/j.knosys.2011.07.021 (дата обращения: 23.05.2025).

4. Permana K. E. Comparison of User Based and Item Based Collaborative Filtering in Restaurant Recommendation System / K. E. Permana. – Modelling, Measurement and Control B, 2024. – Vol. 11, No. 7. – P. 1922–1928. – URL: https://doi.org/10.18280/mmep.110723 (дата обращения: 23.05.2025).

5. Ekstrand M. D., Riedl J. T., Konstan J. A. Collaborative Filtering Recommender Systems / M. D. Ekstrand et al. – Foundations and Trends® in Human–Computer Interaction, 2018. – Vol. 11, No. 3–4. – P. 113–239. – URL: https://doi.org/10.1561/1100000009 дата обращения: 23.05.2025).

6. Lops P., de Gemmis M., Semeraro G. Content-Based Recommender Systems: State of the Art and Trends / P. Lops et al. – In: Recommender Systems Handbook. – Eds. F. Ricci, L. Rokach, B. Shapira. – Springer, 2011. – P. 73–105. – URL: https://doi.org/10.1007/978-0-387-85820-3 (дата обращения: 23.05.2025).

7. Zhao S., King I., Lyu M. R. A Survey of Point of Interest Recommendation in Location Based Social Networks / S. Zhao et al. – arXiv preprint, 2016. – 3 July. – URL: https://doi.org/10.48550/arXiv.1607.00647 (дата обращения: 23.05.2025).

8. Zhang X., Li Y., Wang J. Reliability Aware Task Allocation in Distributed Computing Systems using Hybrid Simulated Annealing and Tabu Search (HSATS) / X. Zhang et al. – Proc. 2012 IEEE 18th International Conference on High Performance Computing and Communications (HPCC). – 2012. – P. 453–460. – URL: https://doi.org/10.1109/HPCC.2012.159 (дата обращения: 23.05.2025).

9. Kirkpatrick S., Gelatt C. D. Jr., Vecchi M. P. Optimization by Simulated Annealing / S. Kirkpatrick et al. – Science, 1983. – Vol. 220, No. 4598. – P. 671–680. – URL: https://doi.org/10.1126/science.220.4598.671 (дата обращения: 23.05.2025).

10. Glover F. Tabu Search: A Tutorial / F. Glover. – Interfaces, 1990. – Vol. 20, No. 4. – P. 74–94. – URL: https://doi.org/10.1287/inte.20.4.74 (дата обращения: 23.05.2025).

11. Metropolis N., Rosenbluth A. W., Rosenbluth M. N., Teller A. H., Teller E. Equation of State Calculations by Fast Computing Machines / N. Metropolis et al. – Journal of Chemical Physics, 1953. – Vol. 21, No. 6. – P. 1087–1092. – URL: https://doi.org/10.1063/1.1699114 (дата обращения: 23.05.2025).

12. Wang B., Bian Z., Mansouri M. Self-adaptive heuristic algorithms for the dynamic and stochastic orienteering problem in autonomous transportation system / B. Wang et al. – Journal of Heuristics, 2023. – Vol. 29. – P. 77–137. – URL: https://doi.org/10.1007/s10732-022-09507-2 (дата обращения: 23.05.2025).

ПРИЛОЖЕНИЕ А
Исходный код программы

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

Рисунок А.1 — QR-код репозитория