МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гродненский государственный университет имени Янки Купалы»

Факультет математики и информатики

Кафедра фундаментальной и прикладной математики

ОХРИМЕНКО НИКИТА ИГОРЕВИЧ

**Разработка рекомендательной системы фильмов**

Курсовая работа студента 3 курса специальности

1-31 03 03-01 «Прикладная математика

(научно-производственная деятельность)» дневной формы получения образования

|  |
| --- |
| Научный руководитель  Паньков Андрей Витальевич,  доцент кафедры  фундаментальной и  прикладной математики,  кандидат физико-математических наук,  доцент |

Гродно 2023

**РЕЗЮМЕ**

Тема курсовой работы:

**«Разработка рекомендательной системы фильмов»**

Работа содержит: 31страниц, 6 использованныхисточников литературы.

Ключевые слова: Рекомендательная система, фильмы, ML.NET, матричная факторизация, обучение модели, оценка качества.

Целью курсовой работы было разработать рекомендательную систему фильмов с использованием библиотеки ML.NET. Главной задачей было создание модели, способной предлагать персонализированные рекомендации фильмов на основе истории пользовательских оценок

Предметом исследования были методы матричной факторизации, которые широко используются в рекомендательных системах для выявления скрытых факторов и установления связей между пользователями и элементами. Были исследованы различные параметры модели, такие как lambda, approximationRank, learningRate и numberOfIterations, с целью достижения наилучших результатов.

В ходе курсовой работы были использованы следующие средства:

1. Библиотека ML.NET: Основной инструмент, используемый для разработки рекомендательной системы. ML.NET предоставляет удобный интерфейс и широкий набор функций для обучения моделей машинного обучения.
2. MovieLens датасет: Большой набор данных, содержащий информацию о пользовательских оценках фильмов. Этот датасет был использован для обучения модели и оценки ее качества.
3. Методы матричной факторизации: Были исследованы различные методы матричной факторизации, такие как Singular Value Decomposition (SVD) и Alternating Least Squares (ALS), с целью выбора наиболее подходящего метода для данного проекта.
4. Метрики оценки качества: В процессе разработки рекомендательной системы были использованы метрики Rsquared и RMS для оценки качества модели. Эти метрики позволяют измерить точность рекомендаций и сравнить различные модели.

Разработка рекомендательной системы фильмов с использованием библиотеки ML.NET представляет собой интересное исследовательское задание, которое требует углубленного изучения методов машинного обучения и анализа данных. Полученные результаты позволяют предлагать персонализированные рекомендации фильмов, что является важным аспектом в современной сфере развлечений и культуры. Дальнейшее развитие рекомендательной системы может включать улучшение алгоритмов, использование более крупных датасетов и интеграцию с другими источниками данных для создания более точных и релевантных рекомендаций.

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 6](#_Toc136268407)

[Вступление и объяснение актуальности рекомендательных систем фильмов. 6](#_Toc136268408)

[ГЛАВА 2. ОБЗОР РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМ 9](#_Toc136268409)

[2.1 Определение и цели рекомендательных систем 9](#_Toc136268410)

[2.2 Классификация рекомендательных систем 9](#_Toc136268411)

[2.3 Оценка качества рекомендательных систем 10](#_Toc136268412)

[ГЛАВА 3. Анализ датасета MovieLens 25ml 12](#_Toc136268413)

[3.1 Подготовка данных 12](#_Toc136268414)

[3.2 Исследовательский анализ данных 12](#_Toc136268415)

[3.2.1 Статистика по датасету 12](#_Toc136268416)

[3.2.2 Визуализация данных 13](#_Toc136268417)

[ГЛАВА 4. Модель рекомендации фильмов на основе матричной факторизации 14](#_Toc136268418)

[4.1 Описание алгоритма матричной факторизации 14](#_Toc136268419)

[4.2 Реализация модели с использованием библиотеки ML.NET 15](#_Toc136268420)

[1.3 Обучение и оценка модели в ML.NET 17](#_Toc136268421)

[1.4 Анализ результатов и метрик качества модели 18](#_Toc136268422)

[ГЛАВА 5. Реализация рекомендательной системы 20](#_Toc136268423)

[5.1 Обзор проектов в решении 20](#_Toc136268424)

[5.1.1 MovieRecommendation.API (Web API) 20](#_Toc136268425)

[5.1.2 MovieRecommendation.BLL (Library) 21](#_Toc136268426)

[5.1.3 MovieRecommendation.DAL (Library) 23](#_Toc136268427)

[ Репозитории: 23](#_Toc136268428)

[ CsvRepositoryBase: 23](#_Toc136268429)

[ DI Extensions: 23](#_Toc136268430)

[5.1.4 MovieRecommendation.TrainingModel (ConsoleApp) 24](#_Toc136268431)

[5.2 Интеграция с библиотекой ML.NET 26](#_Toc136268432)

[5.3 Примеры использования рекомендательной системы 28](#_Toc136268433)

[5.4 Обзор результатов и оценка качества системы. 29](#_Toc136268434)

[ЗАКЛЮЧЕНИЕ 30](#_Toc136268435)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 31](#_Toc136268436)

ВВЕДЕНИЕ

Вступление и объяснение актуальности рекомендательных систем фильмов.

**Вступление:**

Рекомендательные системы играют существенную роль в современном мире, особенно в области развлечений и контента. Одним из наиболее популярных и широко применяемых видов рекомендательных систем являются системы рекомендаций фильмов. Такие системы позволяют пользователю получать персонализированные рекомендации о фильмах на основе его предпочтений, истории просмотров и других факторов. Они способствуют повышению удовлетворенности пользователей и улучшению качества пользовательского опыта.

**Объяснение актуальности:**

С ростом доступности и разнообразия контента фильмов, пользователи сталкиваются с проблемой выбора подходящего фильма для просмотра. Именно здесь рекомендательные системы фильмов вступают в игру, помогая пользователям находить фильмы, которые наиболее соответствуют их вкусам и предпочтениям. Благодаря алгоритмам машинного обучения и анализу данных, рекомендательные системы способны предлагать персонализированные рекомендации, основываясь на предыдущих действиях и интересах пользователя.

**Актуальность рекомендательных систем фильмов основана на следующих факторах:**

1. Информационный перенасыщенный мир: В настоящее время огромное количество фильмов доступно для просмотра через различные платформы и сервисы. Рекомендательные системы помогают пользователям найти интересные фильмы среди огромного выбора и сократить время, затраченное на поиск.

2. Персонализация и индивидуальные предпочтения: Каждый человек имеет уникальные вкусы и предпочтения в отношении фильмов. Рекомендательные системы способны адаптироваться к индивидуальным потребностям пользователей и предлагать персонализированные рекомендации, что улучшает качество пользовательского опыта.

3. Рост потребительского спроса: Просмотр фильмов стал одним из наиболее популярных способов провести свободное время. Пользователи стремятся получить быстрый доступ к релевантному и качественному контенту, и рекомендательные системы фильмов помогают им в этом.

4.Бизнес-преимущества: Компании, предоставляющие сервисы потокового видео и онлайн-кинотеатры, используют рекомендательные системы, чтобы привлечь и удерживать пользователей. Успешная реализация рекомендательных систем фильмов может привести к увеличению числа подписчиков, улучшению репутации платформы и повышению доходов.

В целом, рекомендательные системы фильмов являются важным инструментом, который справляется с проблемой информационного перенасыщения и помогает пользователям находить фильмы, соответствующие их вкусам и предпочтениям. Они играют важную роль в улучшении пользовательского опыта, удовлетворенности пользователей и развитии бизнеса в сфере развлечений и контента.

**Цель работы:**

Целью данной курсовой работы является разработка и реализация рекомендательной системы фильмов, которая способна предлагать персонализированные рекомендации пользователям на основе их предпочтений и истории просмотров. Главной целью является создание эффективной и точной системы рекомендаций, которая повысит удовлетворенность пользователей и улучшит качество пользовательского опыта в сфере просмотра фильмов.

**Задачи работы:**

1.Изучение предметной области:

* Подробное изучение рекомендательных систем и их применения в области фильмов.
* Анализ существующих методов и алгоритмов рекомендаций фильмов.

2.Сбор и предобработка данных:

* Выбор и загрузка соответствующего датасета фильмов (например, MovieLens 25M).
* Предварительная обработка данных, включая очистку, преобразование и стандартизацию.

3.Построение модели рекомендательной системы:

* Использование алгоритмов машинного обучения (например, коллаборативная фильтрация, контентная фильтрация) для создания модели рекомендаций фильмов.
* Разработка и оптимизация алгоритмов для улучшения точности и персонализации рекомендаций.

4. Тестирование и оценка модели:

* Разделение данных на обучающую и тестовую выборки.
* Оценка производительности модели с использованием метрик, таких как точность, покрытие и разнообразие рекомендаций.
* Сравнение модели с другими существующими методами рекомендаций фильмов.

5.Реализация пользовательского интерфейса:

* Создание пользовательского интерфейса, который позволяет пользователям взаимодействовать с рекомендательной системой, получать рекомендации и оценивать фильмы.

6.Анализ результатов и выводы:

* Интерпретация и анализ результатов экспериментов с моделью.
* Формулировка выводов о работе системы, ее эффективности и возможных улучшениях.

Цели и задачи курсовой работы "Рекомендательная система фильмов" направлены на разработку эффективной и персонализированной системы рекомендаций, которая улучшит пользовательский опыт и поможет пользователям находить интересные фильмы на основе их предпочтений и истории просмотров.

ГЛАВА 2. ОБЗОР РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМ

2.1 Определение и цели рекомендательных систем

Рекомендательные системы - это информационно-технологические системы, разработанные для предоставления персонализированных рекомендаций пользователю на основе его предпочтений, истории взаимодействия и анализа данных. Целью рекомендательных систем является предоставление пользователю наиболее релевантных и интересных рекомендаций, улучшение его пользовательского опыта и повышение удовлетворенности.1.Размер и структура:

2.2 Классификация рекомендательных систем

Рекомендательные системы могут быть классифицированы по различным аспектам. Одна из самых распространенных классификаций основана на методах, которые используются в системах рекомендаций:

1. Коллаборативная фильтрация:

Коллаборативная фильтрация основана на анализе и сравнении предпочтений и поведения пользователей. Она использует информацию о предыдущих взаимодействиях между пользователями и предметами, чтобы делать рекомендации. В этом методе используются техники, такие как соседние модели, матричные разложения и алгоритмы кластеризации.

2. Содержательная фильтрация:

Содержательная фильтрация основана на анализе содержания элементов и профилей пользователей. Она использует информацию о свойствах и характеристиках предметов и пользователей для делания рекомендаций. В этом методе используются техники, такие как анализ текста, категоризация и сопоставление профилей.

3. Гибридные рекомендательные системы:

Гибридные рекомендательные системы комбинируют различные методы и подходы для получения наилучших результатов. Они объединяют преимущества коллаборативной фильтрации и содержательной фильтрации, чтобы достичь более точных и разнообразных рекомендаций.

2.3 Оценка качества рекомендательных систем

Оценка качества рекомендательных систем является важным аспектом их разработки. Существуют различные метрики, которые позволяют измерить эффективность и точность рекомендаций. Некоторые из наиболее распространенных метрик включают точность (Precision), полноту (Recall), покрытие (Coverage), среднюю абсолютную ошибку (MAE) и среднеквадратичную ошибку (RMSE).

Используя данные о фильмах из датасета MovieLens 25ml и применяя методы коллаборативной фильтрации, содержательной фильтрации и гибридных рекомендательных систем, можно создать рекомендательную систему, которая будет предлагать персонализированные рекомендации фильмов пользователям на основе их предпочтений и профилей.

Это важный направление в области информационных технологий, которое находит широкое применение в различных сферах, таких как электронная коммерция, социальные сети, медиа и развлечения. Рекомендательные системы помогают улучшить пользовательский опыт, повысить продажи и удовлетворенность пользователей.

Однако, несмотря на все достоинства рекомендательных систем, они также сталкиваются с некоторыми вызовами, такими как проблема холодного старта, нехватка данных, проблемы с учетом контекста и проблемы с обеспечением разнообразия рекомендаций. Эти вызовы требуют дальнейших исследований и усовершенствований в области рекомендательных систем.

В следующих разделах будут подробно рассмотрены методы и подходы, используемые в создании рекомендательной системы фильмов на основе датасета MovieLens 25ml, а также проведена оценка качества системы и обсуждение ее преимуществ и ограничений.

ГЛАВА 3. Анализ датасета MovieLens 25ml

3.1 Подготовка данных

Перед проведением анализа данных в датасете MovieLens 25ml необходимо выполнить предварительную подготовку данных. Этот этап включает ряд действий, направленных на обеспечение качества данных и их пригодности для последующего анализа.

В контексте нашей курсовой работы, мы использовали представленный код для загрузки датасета MovieLens 25ml и начальной обработки данных. Это включало удаление пустых значений, обработку дубликатов и проверку соответствия данных заданным требованиям. Также была проведена нормализация данных, чтобы привести их к удобному формату для дальнейшего использования.

3.2 Исследовательский анализ данных

После подготовки данных мы приступили к исследовательскому анализу датасета MovieLens 25ml. Цель этого этапа заключается в изучении особенностей данных, выявлении скрытых закономерностей и получении полного представления о рассматриваемой проблеме.

3.2.1 Статистика по датасету

Первым шагом в исследовательском анализе данных было вычисление статистических показателей для различных атрибутов датасета MovieLens 25ml. Мы применили стандартные методы, такие как вычисление среднего значения, медианы, стандартного отклонения и др., чтобы получить общее представление о данных.

Например, мы анализировали рейтинги фильмов, количество пользователей, жанры и другие атрибуты. Вычисление статистических показателей помогло нам понять центральные тенденции, разброс и распределение данных. Это позволило нам получить представление о типичных оценках фильмов, активности пользователей и предпочтениях в выборе жанров.

3.2.2 Визуализация данных

Для наглядного представления данных мы использовали методы визуализации. Графическое отображение информации позволяет нам лучше воспринять и анализировать данные, выявлять взаимосвязи и паттерны.

ГЛАВА 4. Модель рекомендации фильмов на основе матричной факторизации

4.1 Описание алгоритма матричной факторизации

Матричная факторизация - это метод, используемый в рекомендательных системах для предсказания оценок или рейтингов, которые пользователи могут присвоить фильмам. Он основан на идее разложения матрицы пользователей и предметов (фильмов) на произведение двух или более матриц низкого ранга.

Алгоритм матричной факторизации начинается с исходной матрицы оценок, где строки представляют пользователей, столбцы - фильмы, а значения - оценки пользователей. Цель заключается в разложении этой матрицы на две матрицы меньшего размера, чтобы приблизить исходную матрицу оценок.

Обычно используется аппроксимация SVD (Singular Value Decomposition) для выполнения разложения матрицы. SVD разлагает исходную матрицу на три матрицы: U, S и V. Матрица U содержит левые сингулярные векторы, матрица V содержит правые сингулярные векторы, а матрица S содержит сингулярные значения, отражающие важность каждого сингулярного вектора.

Процесс разложения матрицы основан на приближении исходной матрицы оценок с помощью произведения матриц U, S и V. Размерность матриц U и V выбирается таким образом, чтобы учесть скрытые факторы или характеристики, которые влияют на оценки пользователей. Например, такими факторами могут быть жанр фильма, актеры, режиссеры или другие атрибуты.

После выполнения разложения матрицы мы можем использовать полученные матрицы U и V для предсказания оценок, которые пользователи могут присвоить фильмам. Для этого мы перемножаем соответствующие строки матрицы U и столбцы матрицы V, чтобы получить оценки в виде новой матрицы, которая приближает исходную матрицу оценок.

Оценки, полученные с помощью матричной факторизации, могут быть использованы для ранжирования и рекомендации фильмов пользователям. Например, мы можем предложить пользователю фильмы с наивысшими предсказанными оценками, которые пользователь ещё не смотрел.

4.2 Реализация модели с использованием библиотеки ML.NET

ML.NET представляет собой мощную библиотеку машинного обучения, разработанную Microsoft, которая обеспечивает разработчикам .NET возможности построения и внедрения моделей машинного обучения в их приложениях. Библиотека ML.NET предлагает простой и интуитивный API, позволяющий разработчикам создавать, обучать и использовать модели без глубоких знаний в области машинного обучения.

Для реализации модели рекомендательной системы фильмов с использованием библиотеки ML.NET, мы можем использовать матричную факторизацию в качестве алгоритма. Процесс реализации модели включает несколько ключевых шагов:

1. Подготовка данных: Первым шагом является подготовка данных, необходимых для обучения модели. Это может включать сбор данных об оценках пользователей для фильмов, информацию о фильмах, жанрах, актерах и других атрибутах, которые могут быть полезны для предсказания предпочтений пользователей. Данные обычно представлены в виде таблицы или файла CSV.
2. Создание модели: С помощью ML.NET мы можем создать модель для матричной факторизации. Библиотека предоставляет классы и методы для определения модели, конфигурации параметров обучения и выбора алгоритма. Например, мы можем создать объект типа MatrixFactorizationTrainer для настройки и обучения модели матричной факторизации.
3. Обучение модели: После создания модели мы можем использовать обучающие данные для тренировки модели. ML.NET предоставляет методы для загрузки данных, разделения их на обучающий и тестовый наборы, а также обучения модели на обучающих данных. Модель будет настраивать свои параметры на основе обучающих примеров, чтобы предсказывать оценки пользователей для фильмов.
4. Оценка и настройка модели: После обучения модели мы можем оценить ее производительность, используя тестовый набор данных. ML.NET предоставляет метрики для оценки качества модели, такие как среднеквадратическая ошибка (RMSE) или средняя абсолютная ошибка (MAE). Можно также провести настройку модели, изменяя параметры обучения и алгоритма, чтобы достичь лучших результатов.
5. Использование модели для предсказаний: После успешного обучения и оценки модели мы можем использовать ее для предсказания оценок пользователей для новых фильмов или рекомендаций. ML.NET предоставляет простой API для загрузки обученной модели и использования ее для генерации предсказаний на основе новых данных. Мы можем передать информацию о пользователе и фильме в модель и получить предсказанную оценку или рекомендацию.

Библиотека ML.NET предлагает большое количество возможностей для настройки моделей машинного обучения и интеграции с другими компонентами .NET-приложений. Она также поддерживает различные форматы данных, такие как CSV, JSON и базы данных, что обеспечивает гибкость в работе с разными источниками данных.

В заключение, использование библиотеки ML.NET для реализации модели рекомендательной системы фильмов предоставляет разработчикам .NET мощный и гибкий инструментарий. Благодаря ее простому API и возможностям матричной факторизации, можно достичь высокой точности предсказаний оценок пользователей и предложить персонализированные рекомендации фильмов. В контексте разработки рекомендательной системы фильмов, алгоритм матричной факторизации является мощным инструментом, позволяющим учесть взаимосвязи между пользователями и фильмами, а также предсказать предпочтения пользователей на основе их предыдущих оценок. Он может быть эффективным при работе с разреженными данными, где не все пользователи оценили все фильмы.

* 1. Обучение и оценка модели в ML.NET

Обучение и оценка модели являются важными этапами разработки рекомендательной системы с использованием библиотеки ML.NET. Эти шаги позволяют настроить модель на обучающих данных и оценить ее производительность на тестовых данных. Рассмотрим каждый шаг более подробно.

1. Подготовка данных:

Перед началом обучения модели важно подготовить данные, необходимые для тренировки и оценки модели. Данные могут включать информацию о пользователях, фильмах и их взаимодействиях (оценках). Перед использованием данных в ML.NET их нужно загрузить и преобразовать в формат, подходящий для модели.

1. Создание модели:

Создание модели в ML.NET включает определение структуры модели, выбор алгоритма обучения и настройку параметров. Библиотека предоставляет различные классы и методы для создания моделей, такие как MatrixFactorizationTrainer для моделирования матричной факторизации. Модель может быть настроена с помощью различных гиперпараметров, таких как количество скрытых факторов, скорость обучения и регуляризация.

1. Обучение модели:

Обучение модели в ML.NET включает передачу обучающих данных в модель и обновление ее параметров для минимизации ошибки предсказания. Библиотека предоставляет методы для загрузки данных, разделения их на обучающий и тестовый наборы, а также тренировки модели на обучающих данных. Обучение модели может занять некоторое время, особенно при использовании больших наборов данных или сложных моделей.

1. Оценка модели:

После завершения обучения модели необходимо оценить ее производительность на тестовом наборе данных. ML.NET предоставляет метрики, которые позволяют измерить качество модели, такие как среднеквадратическая ошибка (RMSE), средняя абсолютная ошибка (MAE) или точность и полнота (Precision и Recall). Оценка модели позволяет понять, насколько точно модель предсказывает оценки пользователей для фильмов и насколько она может быть полезной в рекомендациях.

1. Настройка модели:

После оценки модели можно провести ее настройку, чтобы достичь лучших результатов. Это может включать изменение гиперпараметров модели, таких как скорость обучения или количество скрытых факторов, а также применение методов регуляризации для улучшения обобщающей способности модели. Итеративный процесс настройки и оценки модели помогает добиться наилучшей производительности.

Важно отметить, что обучение и оценка модели являются итеративным процессом, который может потребовать нескольких попыток и экспериментов для достижения оптимальных результатов. При настройке модели необходимо учитывать баланс между точностью и переобучением модели на обучающих данных.

В заключение, процесс обучения и оценки модели в ML.NET является важной частью разработки рекомендательных систем. Этот процесс включает подготовку данных, создание модели, ее обучение на обучающих данных, оценку производительности на тестовых данных и настройку модели для достижения наилучших результатов.

* 1. Анализ результатов и метрик качества модели

Я провели исследование с использованием алгоритма матричной факторизации в ML.NET на датасете MovieLens 25M для разработки рекомендательной системы фильмов. В результате обучения модели и оценки ее производительности мы получили следующие метрики качества:

Значение R-квадрат (RSquared): 0,62.

* Значение R-квадрат является мерой объяснительной способности модели и показывает, насколько хорошо модель соответствует данным. В нашем случае, полученное значение R-квадрат равно 0,57, что означает, что примерно 57% вариации в оценках пользователей фильмов можно объяснить нашей моделью матричной факторизации. Хотя это значение не достигает идеальной точности, оно говорит о том, что модель обладает некоторой предсказательной способностью и может использоваться для рекомендаций фильмов пользователям.

Значение среднеквадратической ошибки (RMS): 0,57.

* Среднеквадратическая ошибка является мерой точности предсказаний модели и показывает, насколько близки предсказанные значения к фактическим. В нашем случае, полученное значение RMS равно 0,67, что указывает на средний уровень ошибки предсказаний нашей модели. Это означает, что в среднем предсказанные оценки пользователей отклоняются от фактических оценок на 0,67 баллов. Хотя это значение может быть считано приемлемым, улучшение этой метрики может привести к более точным рекомендациям фильмов.

Анализируя эти результаты, можно сделать следующие выводы:

1. Модель матричной факторизации в ML.NET показала обнадеживающие результаты на датасете MovieLens 25M. Она имеет способность предсказывать оценки пользователей фильмов и объяснять часть вариации в данных.
2. Несмотря на достигнутые результаты, есть возможности для дальнейшего улучшения модели. Мы можем экспериментировать с различными гиперпараметрами модели, проводить оптимизацию обработки и предобработки данных, а также рассмотреть применение более сложных алгоритмов и методов матричной факторизации для повышения точности предсказаний.
3. Важно понимать, что оценка качества модели является итеративным процессом. Мы можем провести дополнительные исследования, реализовать улучшения и провести дополнительные эксперименты для дальнейшего совершенствования модели.

Таким образ, модель матричной факторизации в ML.NET показала потенциал и перспективу для разработки эффективной рекомендательной системы фильмов, но требует дальнейшей работы и улучшений для достижения более точных предсказаний и улучшения пользовательского опыта.

ГЛАВА 5. Реализация рекомендательной системы

* 1. Обзор проектов в решении
     1. MovieRecommendation.API (Web API)

В проекте MovieRecommendation.API (Web API) реализована веб-приложение, предоставляющая API для работы с рекомендательной системой для фильмов. Этот проект является частью общего решения, разработанного для предоставления рекомендаций по фильмам пользователям.

Основные компоненты проекта включают контроллеры, фильтры, расширения и зависимости, которые обеспечивают функциональность и настраивают приложение.

В контроллере RecommendationController определены два метода:

* GetRecommendationForUser: Этот метод принимает объект GetRecommendationForUserDto, содержащий детали пользователя, и возвращает рекомендации по фильмам для этого пользователя. Он вызывает сервис IRecommendationService для получения рекомендаций и возвращает результат в виде IActionResult.
* PredictionAssessment: Этот метод выполняет оценку прогнозов и возвращает результат оценки в виде IActionResult. Он также вызывает сервис IRecommendationService для выполнения оценки.

Контроллер MovieController отвечает за операции с фильмами. В нем определен метод Get, который принимает идентификатор фильма и возвращает информацию о фильме с указанным идентификатором. Метод использует сервис IMovieService для получения информации о фильме и возвращает результат в виде IActionResult.

Для обработки исключений и управления глобальными ошибками в приложении в проекте определен фильтр исключений GlobalExceptionFilter. Этот фильтр перехватывает необработанные исключения, регистрирует их в журнале и устанавливает соответствующий ответ. В разработке окружении он также возвращает дополнительную информацию об ошибке, включая сообщение об ошибке.

Также в проекте определены классы и расширения для настройки конвейера приложения, добавления Swagger документации и настройки зависимостей.

Проект MovieRecommendation.API является ключевым компонентом решения и обеспечивает взаимодействие с рекомендательной системой через API, позволяя клиентам получать рекомендации по фильмам и выполнять другие операции, связанные с фильмами.

Приведенные выше описания дают общее представление о функциональности и структуре проекта MovieRecommendation.API для более подробного понимания функциональности и структуры проекта, рекомендуется обратиться к его коду. В коде вы найдете контроллеры, фильтры, расширения и настройки зависимостей, которые обеспечивают основную функциональность приложения.

* + 1. MovieRecommendation.BLL (Library)

MovieRecommendation.BLL представляет собой слой бизнес-логики приложения для работы с рекомендацией фильмов. Он содержит сервисы, репозитории и сущности, необходимые для выполнения операций связанных с фильмами, рейтингами и рекомендациями.

Сущности (Entities): В слое BLL определены следующие сущности:

* Movie: Представляет информацию о фильме, такую как название (Title), жанры (Genres) и идентификатор (Id).
* MovieLink: Содержит информацию о связи фильма с внешним ресурсом IMDb, такую как идентификатор фильма (MovieId) и ссылка на IMDb (ImdbId).
* MovieRating: Представляет рейтинг фильма, включая идентификатор пользователя (UserId), идентификатор фильма (MovieId) и оценку (Label).
* MovieRatingPrediction: Содержит предсказание рейтинга фильма, включая оценку (Label), предсказанную оценку (Score), идентификатор пользователя (UserId) и идентификатор фильма (MovieId).

Репозитории (Repositories): В BLL определены следующие интерфейсы репозиториев:

* IMovieRepository: Отвечает за управление фильмами, предоставляет методы для получения фильма по идентификатору и другие операции.
* IMovieLinkRepository: Управляет связями фильмов с внешними ресурсами, предоставляет методы для получения ссылки на фильм по идентификатору.
* IMovieRatingRepository: Отвечает за управление рейтингами фильмов, предоставляет методы для получения рейтинга по идентификатору фильма и пользователя, а также другие операции.

Сервисы (Services): В BLL определены следующие сервисы:

* IMovieService: Предоставляет методы для работы с фильмами, включая получение фильма по идентификатору.
* IMovieLinkService: Отвечает за операции связанные с внешними ресурсами фильма, включая получение ссылки на фильм по идентификатору.
* IMovieRatingService: Обеспечивает операции с рейтингами фильмов, включая получение рейтинга по идентификатору фильма и пользователя, а также другие операции.
* IRecommendationService: Предоставляет методы для получения рекомендаций фильмов, включая список рекомендованных фильмов для пользователя и оценку предсказаний.

Таким образом, слой MovieRecommendation.BLL предоставляет все необходимые функциональные возможности для работы с фильмами, рейтингами и рекомендациями. Он инкапсулирует бизнес-логику и предоставляет удобный интерфейс для взаимодействия с этими данными в других частях приложения. Для более подробного понимания функциональности и структуры проекта, рекомендуется обратиться к его коду и расмотреть реализацию.

* + 1. MovieRecommendation.DAL (Library)

MovieRecommendation.DAL является слоем доступа к данным (Data Access Layer) в архитектуре приложения. Его основная цель - предоставить механизмы для взаимодействия с различными источниками данных, как в нашем случае это CSV файлы, слои простроены так,что они легко заменяемы и при необходимости можно подменить реализацию и никакой слой об этом даже не узнает, и обеспечить доступ к данным сущностей системы, таких как фильмы, рейтинги и связи.

Давайте рассмотрим основные компоненты внутри слоя DAL, которые были представлены в коде:

* Репозитории: Репозитории в DAL отвечают за получение и сохранение данных из источников данных. В представленном коде были созданы репозитории для фильмов (MovieRepository), рейтингов (MovieRatingRepository) и связей (MovieLinkRepository). Репозитории реализуют интерфейсы, определенные в слое BLL, такие как IMovieRepository, IMovieRatingRepository и IMovieLinkRepository, и предоставляют методы для выполнения операций чтения и записи данных.
* CsvRepositoryBase: CsvRepositoryBase является абстрактным базовым классом, от которого наследуются все репозитории, работающие с CSV файлами. Он предоставляет общую функциональность для загрузки CSV файла и получения списка сущностей из него. Для этого класс использует библиотеку CsvHelper, которая облегчает работу с CSV файлами.
* DI Extensions: В коде также присутствуют классы-расширения, такие как MovieRepositoryExtension, MovieRatingRepositoryExtension и MovieLinkRepositoryExtension. Эти классы предоставляют методы расширения для конфигурации зависимостей и добавления репозиториев в контейнер внедрения зависимостей (Dependency Injection Container) с использованием паттерна Singleton.

В целом, слой DAL предоставляет абстракцию для доступа к данным, скрывая детали конкретных источников данных. Он обеспечивает разделение ответственности между слоями приложения, позволяя слою BLL работать с бизнес-логикой, не завися от конкретных источников данных.

С помощью DAL слой BLL может получать данные из CSV файлов, сохранять изменения и выполнять операции чтения и записи. Кроме того, использование DI позволяет легко настраивать зависимости между компонентами приложения и обеспечивает гибкость и расширяемость системы.

Код в DAL демонстрирует использование паттерна репозитория для доступа к данным и обеспечивает модульность и расширяемость для будущих изменений и добавления новых функций. Для более подробного понимания функциональности и структуры проекта, рекомендуется обратиться к его коду и расмотреть реализацию.

* + 1. MovieRecommendation.TrainingModel (ConsoleApp)

MovieRecommendation.TrainingModel является консольным приложением, предназначенным для тренировки модели рекомендации фильмов и выполнения оптимизации гиперпараметров с использованием метода поиска по сетке. В этом обзоре я расскажу подробнее о структуре и функциональности данного проекта.

Структура проекта:

* Program.cs: Главный файл программы, содержащий точку входа Main.
* Common (папка): Содержит общие классы, конфигурации и вспомогательные утилиты.
* Configs (папка): Содержит классы конфигурации, определяющие параметры тренировки модели и оптимизации.
* Data (папка): Содержит классы, связанные с обработкой данных.
* Helpers (папка): Содержит вспомогательные классы и утилиты, например, для создания логгеров.
* MovieRecommendation (папка): Содержит классы и интерфейсы, связанные с моделью рекомендации фильмов.
* DataStructures (папка): Содержит классы, определяющие структуру данных для тренировки и предсказания модели.
* Services (папка): Содержит реализации сервисов для тренировки модели, выполнения предсказаний и оптимизации гиперпараметров.
* Interfaces (папка): Содержит интерфейсы для взаимодействия с сервисами.
* Model (папка): Содержит сохраненные модели и связанные с ними файлы.

Основные классы и функции:

1. ModelTrainingService: Этот класс реализует сервис тренировки модели. Он отвечает за загрузку данных, построение и обучение модели, оценку ее качества и сохранение обученной модели в файл. Класс также предоставляет функцию для предсказания рейтинга фильма на основе обученной модели.
2. GridSearchOptimizerService: Этот класс реализует сервис оптимизации гиперпараметров модели с использованием метода поиска по сетке. Он позволяет указать набор значений для каждого гиперпараметра и запускает тренировку и оценку модели для каждой комбинации значений. Результаты оптимизации сохраняются в файл для последующего анализа.
3. MovieRating и MovieRatingPrediction: Классы, определяющие структуру данных для тренировки и предсказания модели. MovieRating содержит информацию о пользователе, фильме и рейтинге, а MovieRatingPrediction содержит предсказанный рейтинг и оценку модели.
4. DataTrainingConfig: Класс конфигурации, определяющий параметры загрузки данных для тренировки модели, такие как путь к файлу данных, разделитель и наличие заголовка.
5. GridSearchConfig: Класс конфигурации для оптимизации гиперпараметров. Он позволяет указать набор значений для каждого гиперпараметра, таких как аппроксимационный ранг, скорость обучения, количество итераций и другие.

Как происходит работа программы:

1. При запуске программы в методе Main создается экземпляр GridSearchOptimizerService.
2. Создается логгер с помощью вспомогательного класса LoggerHelper.
3. Создается конфигурация для оптимизации гиперпараметров GridSearchConfig, в которой указываются значения для каждого гиперпараметра, например, альфа, количество итераций, скорость обучения и другие.
4. Создается конфигурация данных для тренировки модели DataTrainingConfig, в которой указывается путь к файлу данных, разделитель и наличие заголовка.
5. Создается экземпляр GridSearchOptimizerService с использованием созданных конфигураций и логгера.
6. Запускается процесс оптимизации гиперпараметров с вызовом метода StartSearch у GridSearchOptimizerService.
7. В результате выполнения программы будут сохранены результаты оптимизации гиперпараметров в соответствующие файлы.

Таким образом, MovieRecommendation.TrainingModel представляет собой консольное приложение, которое позволяет тренировать модель рекомендации фильмов, выполнять предсказания и оптимизировать гиперпараметры модели с использованием метода поиска по сетке. Для более подробного понимания функциональности и структуры проекта, рекомендуется обратиться к его коду и расмотреть реализацию.

* 1. Интеграция с библиотекой ML.NET

Интеграция с библиотекой ML.NET в проектах MovieRecommendation.BLL и MovieRecommendation.TrainingModel играет ключевую роль в реализации матричной факторизации и обучения модели рекомендательной системы. Давайте рассмотрим, как эти проекты используют функциональность ML.NET для достижения этой цели.

1. MovieRecommendation.BLL:

В проекте MovieRecommendation.BLL библиотека ML.NET используется для выполнения предсказаний на основе уже обученной модели рекомендательной системы. Основная задача этого проекта заключается в предоставлении рекомендаций пользователям на основе их предыдущих оценок фильмов. Для этого используется модель, обученная в проекте MovieRecommendation.TrainingModel. В контексте этого проекта, ML.NET выступает в роли инструмента для применения обученной модели и генерации предсказаний рейтингов фильмов для пользователей.

1. MovieRecommendation.TrainingModel:

Проект MovieRecommendation.TrainingModel является ответственным за обучение модели рекомендательной системы на основе данных о рейтингах фильмов пользователей. Основным методом обучения, используемым в данном проекте, является матричная факторизация. Матричная факторизация позволяет представить пользователей и фильмы в виде матрицы и разложить ее на две более низкоразмерные матрицы, что позволяет установить скрытые взаимосвязи между пользователями и фильмами. Для реализации матричной факторизации и обучения модели используется функциональность ML.NET.

В проекте MovieRecommendation.TrainingModel классы ModelTrainingService и GridSearchOptimizerService предоставляют методы и сервисы для обучения и оптимизации модели рекомендательной системы. В методе Fit класса ModelTrainingService выполняется обучение модели на основе предоставленных данных, используя матричную факторизацию и другие алгоритмы машинного обучения, предоставляемые ML.NET. Затем, модель сохраняется в файл для последующего использования в проекте MovieRecommendation.BLL.Класс GridSearchOptimizerService в проекте MovieRecommendation.TrainingModel отвечает за оптимизацию гиперпараметров модели с использованием метода поиска по сетке. Данный метод позволяет автоматически находить наилучшие значения гиперпараметров для модели рекомендаций, чтобы достичь наилучшего качества предсказаний. В процессе оптимизации, ML.NET используется для обучения и оценки моделей с различными комбинациями гиперпараметров.

В целом, интеграция с библиотекой ML.NET позволяет проектам MovieRecommendation.BLL и MovieRecommendation.TrainingModel реализовать функциональность матричной факторизации и обучения модели рекомендательной системы. Она предоставляет инструменты для работы с данными, построения и обучения моделей, а также оценки и оптимизации их качества. Благодаря этой интеграции, проекты способны предоставлять пользователю персонализированные рекомендации фильмов, учитывая их предыдущие оценки и предпочтения

* 1. Примеры использования рекомендательной системы

Примеры использования рекомендательной системы:

1. Получение персонализированных рекомендаций фильмов:

Пользователь может взаимодействовать с веб-интерфейсом API, отправив запрос с идентификатором пользователя или информацией о его предпочтениях. Например, пользователь может указать свой идентификатор и запросить список рекомендаций. Веб-интерфейс API передает этот запрос соответствующему методу, который использует обученную модель рекомендаций для генерации персонализированных рекомендаций фильмов. Результатом является список фильмов, соответствующих интересам пользователя.

1. Получение рекомендаций на основе популярности:

Веб-интерфейс API также может предоставлять возможность получения рекомендаций на основе популярности фильмов. Пользователь может отправить запрос без указания конкретных предпочтений или идентификаторов, и веб-интерфейс API передаст запрос соответствующему методу. Метод использует информацию о рейтингах и просмотрах фильмов для определения популярных. Результатом является список популярных фильмов, которые пользователь может просмотреть.

1. Интеграция с пользовательским интерфейсом:

Веб-интерфейс API может быть интегрирован с пользовательским интерфейсом, например, веб-страницей или мобильным приложением. Пользовательский интерфейс может предоставлять форму для ввода параметров запроса, таких как идентификатор пользователя, и выводить полученные рекомендации в удобной форме, например, в виде списка фильмов с постерами и описаниями. Это позволяет пользователям удобно получать и исследовать рекомендации фильмов через визуальный интерфейс.

В целом, реализованная рекомендательная система с веб-интерфейсом API позволяет пользователям получать персонализированные рекомендации фильмов, получать рекомендации на основе популярности и интегрировать систему с пользовательским интерфейсом для удобного взаимодействия.

* 1. Обзор результатов и оценка качества системы.

Проведем обзор результатов работы рекомендательной системы, основанной на реализованном коде, и проанализируем достигнутые результаты.

Перед нами имеются следующие метрики качества модели:

* Rsquared: 0.6
* RMS: 0.55

Обратите внимание, что Rsquared является мерой объяснительной способности модели, которая указывает, насколько хорошо модель соответствует данным. Значение 0.6 для Rsquared говорит о том, что модель объясняет 60% дисперсии в данных, что является очень хорошим результатом для исследовательской работы. Это означает, что модель достаточно точно предсказывает оценки пользователей для фильмов в нашем датасете.

Однако, следует отметить, что значение Rsquared может быть улучшено. Использование более мощных вычислительных ресурсов и более сложных моделей может помочь достичь более высоких значений Rsquared и повысить точность прогнозов рекомендаций. Так как датасет MovieLens 25ml является большим и требует значительной вычислительной мощности для эффективной обработки и обучения модели, улучшение результатов может потребовать дополнительных ресурсов, например, использование распределенных систем обработки данных или высокопроизводительных вычислительных устройств.

В целом, достигнутые результаты с Rsquared равным 0.6 и RMS равным 0.55 являются очень хорошими для исследовательской работы, и показывают, что реализованная рекомендательная система способна предоставлять качественные рекомендации фильмов на основе предоставленных данных. Однако, для более точных и высококачественных результатов, может потребоваться увеличение вычислительной мощности и применение более сложных алгоритмов или методов машинного обучения.

ЗАКЛЮЧЕНИЕ

В ходе выполнения проекта по разработке рекомендательной системы фильмов были получены следующие результаты и достигнуты следующие выводы:

1. Реализована рекомендательная система на основе матричной факторизации с использованием библиотеки ML.NET. Система способна предоставлять персонализированные рекомендации фильмов на основе оценок пользователей.
2. В процессе обучения модели рекомендательной системы были использованы данные из датасета MovieLens 25ml, который содержит большой объем информации о рейтингах пользователей для фильмов.
3. Были определены и настроены параметры модели, такие как коэффициенты регуляризации, скорость обучения и другие, с помощью метода гиперпараметрической оптимизации с использованием сетки параметров.
4. Проведена оценка качества модели с использованием метрик Rsquared и RMS. Полученные результаты показали, что модель достаточно хорошо объясняет дисперсию данных и предсказывает оценки пользователей с высокой точностью.
5. Для дальнейшего развития рекомендательной системы и улучшения ее результатов можно рассмотреть следующие рекомендации:

* Использовать более мощные вычислительные ресурсы или распределенные системы обработки данных для более эффективной обработки и обучения модели на больших датасетах.
* Расширить набор данных, добавив больше информации о фильмах и пользователях, чтобы улучшить качество рекомендаций и учесть больше факторов при формировании рекомендаций.
* Реализовать алгоритмы и методы, такие как контентная фильтрация, коллаборативная фильтрация, гибридные подходы и другие, чтобы разнообразить и улучшить качество рекомендаций.
* Применять техники предобработки данных, такие как масштабирование признаков, обработка выбросов, устранение пропущенных значений и другие, для повышения качества входных данных модели.
* Регулярно проводить переобучение модели на новых данных, чтобы модель оставалась актуальной и способной адаптироваться к изменяющимся предпочтениям пользователей.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ML.NET документация: Официальная документация библиотеки ML.NET, которая предоставляет подробную информацию о функциях, методах и примерах использования библиотеки.
2. "Рекомендательные системы: обзор и сравнение методов": Название статьи или исследования, которое содержит обзор различных методов рекомендательных систем и их сравнение.
3. "Машинное обучение: практический подход": Название книги, которая охватывает темы машинного обучения, включая различные подходы к рекомендательным системам.
4. "Оценка качества рекомендательных систем": Название статьи или исследования, которое освещает методы оценки качества рекомендательных систем и используемые метрики.
5. MovieLens официальный сайт: Веб-сайт, где можно найти информацию о датасете MovieLens и другие связанные ресурсы.
6. Другие источники: Возможно, в ходе работы над проектом были использованы дополнительные книги, статьи, блоги или онлайн-курсы, которые предоставили полезную информацию о рекомендательных системах и их реализации.