САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

КАФЕДРА КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ И МНОГОПРОЦЕССОРНЫХ СИСТЕМ

Кокачев Виктор Алексеевич

Магистерская диссертация

Рекомендательные системы в контексте технологий больших данных

Направление 020402

Фундаментальная информатика и информационные технологии

Магистерская программа “Вычислительные технологии”

Научный руководитель,
кандидат физ.-мат. наук,
доцент
Корхов В. В.

Санкт-Петербург

2018

Содержание

Введение

Последние десятилетия характеризуются бурным развитием сети Интернет: ежедневно в глобальной паутине генерируются и накапливаются колоссальные объемы информации. Пользователю приходится работать с этими данными: обрабатывать, систематизировать, находить релевантные для его информационных потребностей данные. Человеку сложно отобрать интересующую его информацию путем обычного просмотра, так как часто релевантная информация теряется среди больших объемов данных. В связи с этим, создаются инструменты, которые могут помочь человеку в поиске, предлагая тот контент, который предпочтителен для пользователя. Такие программные средства получили название рекомендательные системы.

Рекомендательные системы (*англ. recommender systems*) - программы и сервисы, которые анализируют интересы пользователей и пытаются предсказать, что именно будет наиболее интересно для конкретного пользователя в данный момент времени. Такие системы показывают предпочтительность контента для конкретного юзера на основе данных, указанных пользователем явно или на основе его взаимодействия с системой. Рекомендательные системы должны обладать следующими свойствами: система должна адаптироваться под конкретного пользователя, так как предпочтения могут значительно отличаться у разных людей; система должна учитывать текущие предпочтения пользователя, подстраиваясь под него со временем; система должна постоянно находить новые области информации и предлагать их пользователю. Все это делает ресурсы, основанные на рекомендательных механизмах, привлекательными для пользователя. С другой стороны, подобные системы интересны и владельцам самих ресурсов, на которых размещаются рекомендательные системы, так как с

помощью подобных инструментов повышается привлекательность самого ресурса и его контента.

Рекомендательные системы нашли свое применение во многих сферах жизнедеятельности человека: поиске фильмов и научных статей, розничной торговле, социальных сетях, электронной коммерция, онлайн-банкинге и т.д. Подобная задача применима и к сфере музыкальных рекомендаций. В наше время существует большое количество музыкальных сервисов, на которых размещается еще более внушительное число самих музыкальных композиций. Все это делает задачу ручного поиска и прослушивания музыкальных композиций долгой и ресурсоемкой. Поэтому данная задача приобретает все большую актуальность: зачастую пользователь музыкальных сервисов хочет, чтобы система сама предлагала привлекательные для пользователя композиции.

Работа имеет структуру, состоящую из вводного раздела, трех основных глав и заключительной части. В первой главе приведен предметный обзор области рекомендательных систем. Во второй главе освещена практическая часть работы, проведены исследования алгоритмов. Третья глава посвящена реализации рекомендательной системы, показаны результаты работы и оценки ее точности.

Постановка задачи

Цель:

Целью работы ставится разработка рекомендательной системы музыкальных композиций.

Задачи:

- Исследовать виды рекомендательных систем и принципы их построения.
- Спроектировать рекомендательную систему на основе выбранного подхода.
- Реализовать рекомендательную системы музыкальных композиций, оценить качество ее работы.

Обзор литературы

Задача, стоящая в основе рекомендательных систем, возникла сравнительно недавно, однако даже за такой короткий промежуток времени было созданию большое количества работ и статей, связанных с тематикой рекомендательных алгоритмов.

Рекомендательные системы стали важной темой в исследованиях в начале 90-ых, когда появились первые работы по коллаборативной фильтрации [1, 2, 3]. Сам термин “рекомендательные системы на основе коллаборативной фильтрации” был впервые использован Дэвидом Голдбергом в 1992 году в статье «Using collaborative filtering to weave an information tapestry» [4] в процессе работы над рекомендательной системой Tapestry для компании Xerox. Основой для работ по фильтрации на основе контента можно считать [5, 6].

В последующие годы были созданы следующие фундаментальные работы:

Recommender Systems Handbook [7] - один из наиболее известных справочников по рекомендательным системам. В этой работе было систематизировано все многообразие различных методологий и концепций, относящихся к рекомендательным системам из различных прикладных областей: анализ данных, системы принятия решений, маркетинг, статистика. Источник также содержит практическое применение подходов, которые используют в своих рекомендательных системах такие крупные корпорации как Amazon, Google, AT&T.

Еще одним важным трудом в области рекомендательных систем является работа Recommender Systems: The Textbook [8]. В работе освещены различные фундаментальные алгоритмы и методы оценки точности их работы. В книге всесторонне затрагивается тема рекомендательных методик для различных прикладных областей, включая социальные системы, рекомендации новостей и

онлайн-рекламу. Кроме того, освещены и практические аспекты создания рекомендательных систем: вопросы надежности и защиты, ранжирования результатов.

Начало исследованиям в области музыкальных рекомендаций было положено в 2001 году работой [6]. Традиционные музыкальные рекомендательные системы используют совместную фильтрацию или фильтрацию на основе контента [3]. Помимо уже стандартных подходов коллаборативной фильтрации, характерных для всех типов рекомендательных систем, здесь стоит обратить внимание на работы, основанные на гибридных подходах [9], создании плейлистов [10], построении музыкальных социальных сетей [11], тегировании [12].

Глава 1. Обзор предметной области

Рекомендательные системы — программные средства, которые пытаются предсказать какие объекты (фильмы, музыка, книги, новости, веб-сайты и т. д.) будут интересны пользователю, если имеется определенная информация о его предпочтениях [13, 14]. Рекомендации формируются отдельно для каждого человека на основе прошлой активности. Кроме того, имеет значение и поведение остальных пользователей системы.

Существует два основных подхода к построению рекомендаций [15, 16]:

- На основе коллаборативной фильтрации (*англ. collaborative filtering*), которая использует информацию о поведении пользователей в прошлом, например, перечень покупок или оценок объектов, сделанных ранее на сайте интернет-магазина пользователями из той же группы интересов.
- На основе фильтрации содержимого (*англ. content-based information filtering*), при этом в системе содержатся профили, включающие личную информацию пользователей: социальный статус, возраст, место проживания, род деятельности, а также характеристики, выражающие интерес пользователя к объекту; профили объектов интереса включают характеристики, интересующие пользователя.

§1 Развитие рекомендательных систем

Несмотря на то что предметная область рекомендательных систем зародилась совсем недавно, теоретические основы были заложены много лет назад в области машинного обучения. В 50-е годы в этой области науки были сформулированы математические подходы и описаны модели самообучающихся алгоритмов, которые до сих пор лежат в основе всех решений.

В начале 1990-ых коллаборативная фильтрация стала применяться как решение для борьбы с избыточной информацией в вебе [17]. Tapestry (экспериментальный почтовый сервис) [4] стал одной из первых систем, использующих данный подход: она позволяла пользователю создавать вручную запросы, основанные на мнениях или действиях других пользователей (“give me all the messages forwarded by John”). Такие манипуляции требовали определенных действий от пользователей, но с другой стороны, это позволяло пользователям определить актуальность переписки для себя, опираясь на реакцию других участников переписки.

В дальнейшем стали появляться системы фильтрации, которые автоматически определяли релевантные мнения и обобщали их для представления рекомендаций. В программном компоненте GroupLens [2] использовался данный метод для нахождения статей в сети Usenet, которые могли бы быть интересны конкретному пользователю. Пользователям предлагалось оценить статьи, а система, в свою очередь, объединяла их с оценками других пользователей для предоставления персонализированных результатов.

В то же время рекомендательные системы стали предметом повышенного интереса в вопросах взаимодействия человека и компьютера, а также в области машинного обучения и информационного поиска. Вследствие этого,

рекомендательные системы все чаще стали находить применение в областях музыкальных [3], кино- [1] и множества других рекомендаций. За пределами ИТ-индустрии рекомендательные системы все чаще стали появляться в сфере маркетинга как один из способов увеличения числа продаж.

В конце 1990-ых стали появляться коммерческие рекомендательные продукты. Пожалуй, самым известным интегратором подобных технологий того времени является Amazon.com. Основываясь на истории покупок, истории просмотров и текущем просматриваемом товаре, система делала предположения о тех продуктах, которые могли бы быть интересны пользователю. После успеха Amazon многие другие представители сегмента электронной коммерции обратили свое внимание на рекомендательные решения, а некоторые ИТ-компании как NetPerceptions и Strands построили свой бизнес вокруг внедрения систем основанных на рекомендациях в крупные онлайн-магазины.

В 2006 году рекомендательные алгоритмы привлекли к себе дополнительное внимание, когда компания Netflix запустила соревнование Netflix Prize, целью которого было создание алгоритма рекомендаций, который смог бы улучшить результат действующего внутреннего алгоритма CineMatch в тестах на 10%. Это вызвало шквал активности, как в академических кругах, так и среди любителей. Объявленная победителям премия в размере одного миллиона долларов демонстрирует ценность, которая кроется в изучении рекомендательных алгоритмов для крупных компаний.

В то же время компания Google представила свою систему персонализации новостей Google News [19] на основе истории кликов пользователей. В данном случае новостные статьи рассматриваются как объекты интереса, а клики пользователей как присвоение положительного рейтинга новостной статье. Алгоритм коллаборативной фильтрации применяется к собранным рейтингам и

на основе этого делается вывод о персонализированной выдаче статей для конкретного пользователя.

Первое десятилетие 21-го века ознаменовалось бурным ростом социальных сетей, не обошли столь популярное явление и рекомендательные алгоритмы. Facebook - крупнейшая социальная сеть, одной из первых внедрила алгоритм рекомендаций потенциальных социальных связей. Такие рекомендации несут несколько иные цели, чем рекомендации продукта: социальные сети в значительной степени зависят от их роста для увеличения доходов от рекламы, поэтому рекомендации потенциальных друзей обеспечивают быстрый рост и связность сети. Эта задача также называется прогнозированием ссылок в области анализа графов сетей. Поэтому природа данного типа алгоритмов несколько отлична от стандартных рекомендательных алгоритмов.

§2 Обзор существующих решений

Рекомендательные системы уже интегрированы во множество веб-приложений, которые широко используются каждый день миллионами пользователей. Рассмотрим примеры крупнейших ресурсов, использующие рекомендательные механизмы.

LinkedIn - бизнес-ориентированная социальная сеть. Встроенный рекомендательный механизм предлагает пользователю рекомендации людей, которых он, возможно, знает, вакансий, которые могли бы его привлечь, групп, в которые он мог бы захотеть вступить, компаний, которыми он мог бы заинтересоваться. Специализированная система коллаборативной фильтрации LinkedIn основана на технологии Apache Hadoop.

Amazon - одна из крупнейших площадок интернет-торговли — использует рекомендации на основе контента. Когда посетитель выбирает для покупки

какой-либо товар, Amazon на основе этого исходного товара рекомендует посетителю другие товары, приобретенные другими пользователями (с помощью матрицы покупки следующего товара на основе его схожести с предыдущей покупкой). Компания Amazon запатентовала этот подход под названием item-to-item collaborative filtering (коллаборативная фильтрация от элемента к элементу).

Среди сервисов, которые основываются на музыкальных рекомендациях можно выделить следующие:

Last.fm создает музыкальную «станцию» рекомендованных песен, наблюдая, какие группы и отдельные треки пользователь прослушивает на регулярной основе. Last.fm воспроизводит дорожки, которые не присутсвуют в библиотеке пользователя, но часто воспроизводятся другими пользователями с аналогичными интересами. Поскольку этот подход использует поведение пользователей, он является примером совместной фильтрации.

Pandora использует метаданные песен и исполнителей порядка 400 атрибутов, предоставленных проектом Music Genome Project, чтобы сгенерировать «станцию», которая воспроизводит музыку с похожими свойствами. Кроме того, для уточнения результатов «станции» используется обратная связь от пользователя, которая обесценивает определенные атрибуты, когда пользователю не понравилась определенная песня и увеличивает вклад других атрибутов, когда пользователю нравится песня. Данный сервис использует контент-ориентированный подход.

§3 Классификация рекомендательных систем

В базовых подходах для рекомендательных систем могут использоваться два вида данных:

- Информация о взаимодействии пользователей с объектами интереса
- Информация, предоставленная самими пользователями, например, атрибуты, указанные в профиле или релевантные ключевые слова.

Первую группа методов чаще всего называют методами коллаборативной фильтрации, для методов второй группы обычно используется название рекомендаций на основе контента. Еще один тип систем рекомендаций - системы рекомендаций, основанные на знаниях: здесь рекомендации основаны на явно указанных пользовательских требованиях. Некоторые рекомендательные системы могут объединять перечисленные выше аспекты; такие системы называются гибридными. Они сочетают в себе сильные стороны различных подходов для создания методов, которые могут работать более эффективно в узкопрофильных системах.

3.1 Рекомендательные системы на основе коллаборативной фильтрации

Коллаборативная фильтрация (*англ. Collaborative filtering*) вырабатывает рекомендации, основанные на модели предшествующего поведения пользователя. Эта модель может быть построена исключительно на основе поведения данного пользователя или — что более эффективно — с учетом поведения других пользователей со сходными характеристиками. [14, 17, 20]

Коллаборативная фильтрация использует два разных типа входных данных (рис. 1): множество пользователей и множество объектов интереса. Отношения между пользователями и объектами интереса обычно выражаются при помощи

оценок, предоставляемых пользователями, и использующихся в последующих сессиях для прогнозирования оценок, которые пользователь (в нашем случае пользователь $П_a$) мог бы поставить неоцененным объектам интереса. Если предположить, что пользователь $П_a$ в настоящее время взаимодействует с коллаборативной системой рекомендаций, первым делом система должна идентифицировать ближайших соседей (пользователей с аналогичным поведением, как и $П_a$) а затем экстраполировать из рейтингов похожих пользователей рейтинг пользователя $П_a$.

Два основных подхода к коллаборативной фильтрации - это user-based коллаборативная фильтрация и item-based коллаборативная фильтрация. Оба варианта предсказывают, в какой степени пользователь будет интересоваться объектами, которые до сих пор не были им оценены. User-based фильтрация идентифицирует k ближайших соседей активного пользователя и на основе этих ближайших соседей вычисляет прогноз пользователя для определенного объекта интереса. В отличие от user-based фильтрации, при коллаборативной фильтрация на основе элементов для текущего объекта ищутся “соседи”, которые получили аналогичные рейтинги.

Рис.1. Коллаборативная фильтрация

К плюсам данного подхода можно отнести теоретически высокую точность, к минусам: высокий порог входа - не зная ничего об интересах пользователя, рекомендации практически бесполезны.

3.2 Рекомендательные системы, основанные на контенте

Фильтрация на основе контента (*англ. Content-based filtering*) основана на предположении о том, что интересы пользователя постоянны в течение времени. Например, пользователи, заинтересованные в конкретной теме, как правило, не меняют не меняют свои интересы каждый день и будут заинтересованы в данной теме в ближайшем будущем [21].

Фильтрация на основе контента (рис. 2) в качестве входных данных содержит множество пользователей и множество категорий (или ключевых слов), которыми были помечены объекты интереса. Задача систем рекомендаций, основанных на контенте - вычислить множество объектов, которые наиболее близки к категориям, которыми интересуется текущий пользователь ($П_a$).

Основным подходом к фильтрации на основе содержимого является сравнение уже просмотренных пользователем объектов с новыми объектами, которые потенциально могут быть рекомендованы пользователю. Базовым механизмом для определения такого сходства является извлечение ключевых слов непосредственно из контекста, содержащего объект интереса или из метаинформации, которой был проаннотирован объект информации. Подобные подходы к извлечению ключевых слов рассматриваются в работе [22].

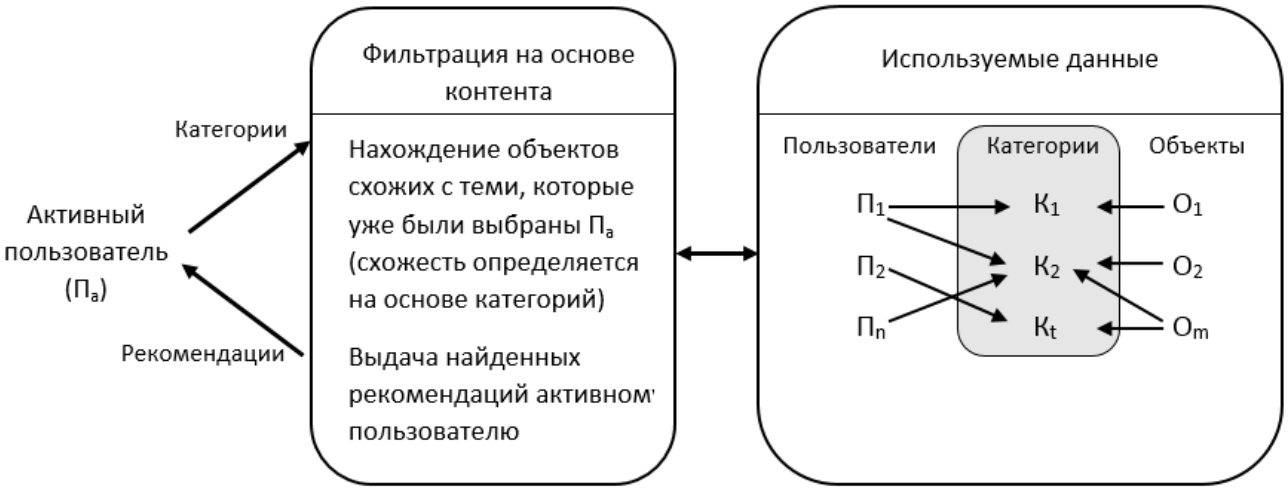

Рис. 2. Фильтрация на основе контента

Плюсами данного подхода является то, что можно давать рекомендации даже незнакомым пользователям, тем самым вовлекая их в сервис, появляется возможность рекомендовать те объекты, которые еще не были никем оценены. К минусам можно отнести более низкую точность, возросшую скорость разработки.

3.3 Рекомендательные системы, основанные на знаниях

По сравнению с подходами, основанными на коллаборативной фильтрации и фильтрации на основе контента, рекомендации, основанные на знаниях (*англ. knowledge-based recommender systems*), в основном не зависят от оценки объектов или их описания с помощью метаданных, а на более глубоких правилах для выявления объектов интереса. Иногда предыдущий подход (content-based) определяют как частный случай knowledge-based, где в качестве знаний выступает информация об объектах интереса, но из-за большой распространенности систем на основе контента последние обычно выносят в

отдельный тип. Дополнительные знания позволяют рекомендовать объекты, не полагаясь на «похожесть» чего-либо, а использовать более сложные условия.

Рекомендации, основанная на знаниях (рис. 3), опираются на следующие входные данные: (a) множество правил (ограничений) или метрик схожести и (b) множество объектов интереса. В зависимости от заданных требований пользователя, правила описывают, какие объекты должны быть рекомендованы. Текущий пользователь $П_a$ формулирует свои предпочтения в терминах свойств элемента, которые, в свою очередь, представляются с точки зрения правил (ограничений). Подробный обзор механизмов вынесения решений, которые могут использоваться в фильтрах подобного рода описаны в [15, 16, 22].

Рис 3. Рекомендации, основанные на знаниях

В качестве плюсов можно отметить возможность исключения рекомендаций уже не актуальных для данного пользователя объектов, минусы - высокая сложность построения и сбора данных.

3.4 Гибридные рекомендательные системы

Помимо трех основных подходов к фильтрации, используется гибридный подход (*англ. hybrid recommender system*), который объединяет возможности

базовых типов. Использование гибридных алгоритмов позволяет достичь более высокой точности.

Применяют различные стратегии построения гибридных классификаторов:

- Взвешенная стратегия

Спрогнозированная оценка для объекта рассчитывается, как взвешенное среднее арифметическое оценок, спрогнозированных различными алгоритмами. Соответствующий пример показан в таблице 1, где для каждого объекта суммируются оценки, полученные с помощью коллаборативной фильтрации и фильтрации, основанной на контенте. Объект O_8 получает наибольший суммарный балл (9,0) и самое высокое место при ранжировании.

Объекты	**O_1**	**O_2**	**O_3**	**O_4**	**O_5**	**O_6**	**O_7**	**O_8**	**O_9**	**O_{10}**
Оценка (O_i, коллаборативная фильтрация)	1.0	3.0	–	5.0	–	2.0	–	4.0	–	–
Оценка (O_i, фильтрация на основе контента)	–	1.0	2.0	–	–	3.0	4.0	5.0	–	–
Суммарный балл (O_i)	1.0	4.0	2.0	5.0	0.0	5.0	4.0	9.0	0.0	0.0
Ранг (O_i)	7	4	6	2	8	3	5	1	9	10

Табл. 1. Пример взвешенной гибридной фильтрации

- Смешанная стратегия

Смешанная гибридная стратегия основана на идее, что прогнозы отдельных рекомендации отображаются в одном интегрированном результате. Например, результаты коллаборативной фильтрации и фильтрации по содержанию могут быть ранжированы как показано в таблице 2. Оценочные баллы могут быть

определены следующим образом: объект с самым высоким совместным прогнозом фильтрации значение получает самый высокий общий балл (10,0), элемент с наилучшей фильтрацией на основе контента значение предсказания получает второй лучший общий балл и т. д.

Объекты	**O_1**	**O_2**	**O_3**	**O_4**	**O_5**	**O_6**	**O_7**	**O_8**	**O_9**	**O_{10}**
Оценка (O_i, коллаборативная фильтрация)	1.0	3.0	–	5.0	–	2.0	–	4.0	–	–
Оценка (O_i, фильтрация на основе контента)	–	1.0	2.0	–	–	3.0	4.0	5.0	–	–
Суммарный балл (O_i)	4.0	8.0	5.0	10.0	0.0	6.0	7.0	9.0	0.0	0.0
Ранг (O_i)	7	3	6	1	8	5	4	2	9	10

Табл. 2. Пример смешанной гибридной фильтрации

- Каскадная стратегия

Каскадная стратегия является итеративным методом построения рекомендательных систем. Первый алгоритм играет роль грубого фильтра, а все следующие алгоритмы корректируют оценки.

§4 Сравнение базовых подходов

Три основных подхода к построению рекомендательных систем используют разную базовую входную информацию и имеют различные сильные и слабые стороны (табл. 3).

Для реализации коллаборативной фильтрация (CF), а также фильтрации на основе контента (CBF) необходима только базовая информация о предмете, например, его название или другие атрибуты, тогда как решения основанные на знаниях требуют более детальной информации о свойствах предмета (а во многих случаях также дополнительных условий и ограничений). CF и CBF более адаптивны в том понимании, что вновь внесенные пользователем оценки автоматически учитываются при будущих запусках алгоритма рекомендации. Напротив, правила в рекомендательных системах, основанных на знаниях необходимо каждый раз адаптировать вручную под вновь внесенные данные.

Свойство интуитивности можно интерпретировать как некую случайность нахождения релевантных объектов, даже когда пользователь не инициировал соответствующего поиска. Такой эффект, в первую очередь, может быть достигнут при использовании подходов коллаборативной фильтрации. В связи с тем, что фильтрация по содержимому не учитывает предпочтений других пользователей, подобное свойство не может быть достигнуто. Подобный эффект для систем, основанных на знаниях, в принципе возможен, однако, его возможность достижения сильно зависит от навыков инженера знаний (который может предвидеть такие свойства при создании рекомендательных правил).

Термин «проблема холодного старта» относится к ситуации, когда возникает необходимость предоставления начальных оценок до того, как алгоритм сможет определить релевантные рекомендации. Такая проблема характерна как для алгоритмов коллаборативной фильтрации, так и для рекомендаций на основе содержимого. Пользователи при коллаборативном подходе должны оценивать набор элементов, прежде чем алгоритм сможет определить ближайших соседей. При использовании рекомендательных алгоритмов на основе контента пользователь также должен указать интересные

ему объекты до того момента, когда алгоритм будет способен определять элементы, похожие на уже оцененные пользователем.

Наконец, свойство прозрачности определяет степень того, насколько понятно можно обосновать результат работы рекомендательных алгоритмов для пользователей. Подобные интерпретации результатов в системах коллаборативной фильтрации полагаются исключительно на механизм “ближайших соседей”, то есть, пользователи, которые интересовались объектом X, также интересуются объектом Y. Алгоритмы, основанные на фильтрации содержимого объясняют свои рекомендации в терминах схожести рекомендуемого элемента с объектами, которыми интересовался пользователь: мы рекомендуем Y, поскольку вы интересовались X, который очень похож на Y. В отличие от прошлых методов, подходы, основанные глубоких знаниях готовы предоставить развернутые пояснения, которые учитывают семантически знания о предметах. Пример такого пояснения - вывод, объясняющий причины того, почему определенный набор требований не позволяют получить конкретный объект в качестве выдачи рекомендательного алгоритма.

Как правило, алгоритмы коллаборативной фильтрации и фильтрации, основанной на содержимом используются для рекомендации контента низкой степени участия такого как фильмы, книги и новостные статьи. Под низкой степенью участия будем понимать то, что вклад неправильной рекомендации довольно низок, поэтому пользователи прикладывают меньше усилий к оценке предмета. Напротив, системы, основанные на знаниях, обычно используются для рекомендаций высокой степени участия таких как финансовые услуги, автомобили и квартиры. В последнем случае, оценки выставляются с низкой частотой, что делает эти предметные области менее доступными для первых двух подходов. Например, пользовательские предпочтения относительно автомобилей могут значительно измениться в течение нескольких лет, при этом они не будут

обнаруженными рекомендательной системой. В то же время, такие сдвиги предпочтений определяются первыми двумя подходами, в связи с тем, что покупки в их предметных областях происходят чаще и, как следствие, соответствующие рейтинги доступны для вынесения рекомендаций.

Подход	Коллаборативная фильтрация	Фильтрация на основе контента	Фильтрация на основе знаний
Быстрое развертывание	Да	Да	Нет
Адаптивность	Да	Да	Нет
Интуитивность	Да	Нет	Нет
Холодный старт	Да	Да	Нет
Прозрачность	Нет	Нет	Да
Высокая степень участия	Нет	Нет	Да

Табл. 3. Сравнение характеристик основных рекомендательных подходов

Глава 2. Проектирование рекомендательной системы

Стандартным методом при реализации рекомендательных систем является подход на основе коллаборативной фильтрации. Это обусловлено тем, что современные рекомендательные системы должны обрабатывать колоссальные объемы данных. Для того, чтобы эффективно, а главное своевременно, выносить рекомендательный вердикт, приходится использовать алгоритмы, которые легко масштабируемы и способны манипулировать большими данными. Такими свойствами и обладает ряд алгоритмов, относящихся к коллаборативному подходу. В данной главе рассмотрим более подробно разные реализации данного подхода и методы их оценки.

§1 Обзор алгоритмов

1.1 User-based

Алгоритм user-based коллаборативной фильтрации можно разделить на три последовательных шага:

1. Необходимо рассчитать сходство между активным пользователем и остальными пользователями («соседями»).
2. Выбрать подмножество пользователей с наиболее высокой схожестью с активным пользователем
3. Вычислить прогноз с использованием оценок соседа.

С момента начала использования user-based алгоритма, были рассмотрены и различным образом скомбинированы возможные методы решения каждого из этих трех этапов. Таким образом, user-based подход считается совокупностью

различных семейств алгоритмов - каждый из них представляет различные стратегии для каждого шага.

Вычисление схожести между пользователями

- На основе корреляции Пирсона.

Один из наиболее старых подходов, однако до сих пор популярный:

$$s(a,u) = \frac{\sum_{i \in I_a \cap I_u}(v_{ai} - \overline{v_a})(v_{ui} - \overline{v_u})}{\sqrt{\sum_{i \in I_a \cap I_u}(v_{ai} - \overline{v_a})^2 \sum_{i \in I_a \cap I_u}(v_{ui} - \overline{v_u})^2}}$$

- Косинусная мера схожести.

В рамках данного подхода пользователи представляются как вектора, состоящие из оценок объектов. Результат, близкий к единице, говорит о близости пользователей, близкий к нулю - об обратном.

$$s(a.u) = \sum_{j \in I} \frac{v_{aj}}{\sqrt{\sum_{k \in I_a} v_{ak}^2}} \frac{v_{uj}}{\sqrt{\sum_{k \in I_u} v_{uk}^2}}$$

- На основе среднеквадратичной ошибки.

Схожесть для двух пользователей рассчитывается как средняя разница между векторами объектов, которые оценили оба пользователя:

$$msd(a,u) = \frac{\sum_{i \in I_a \cap I_u}(v_{ai} - v_{ui})^2}{|I_a \cap I_u|}$$

Затем, те пользователи у которых разница больше, чем некоторый порог L, отбрасываются, а схожесть оставшихся пересчитывается по формуле:

$$s(a,u) = \frac{L - msd(a,u)}{L}$$

Отбор соседей

- Порог сходства.

Отбираем только тех пользователей, чья мера сходства с активным пользователем превосходит некоторый порог.

- Максимальное число соседей.

Выбираем только N пользователей наиболее схожих с активным пользователем.

Вычисление предсказаний

- На основе весов

Вклад каждого соседа взвешенно учитывается при расчете оценок активного пользователя. Чем больше пользователь схож с активным пользователем, тем более явно учитывается его оценка.

- На основе стандартизированной оценки (z-score)

Прежде, чем находить взвешенную оценку пользователя на основе схожести, ее нужно нормализовать. Под нормализацией будем полагать, что рейтинги каждого пользователя принадлежат различным распределениям: существуют пользователи, которые дают только плохие оценки плохим предметам; другие ставят только хорошие оценки для всех предметов, третьи; которые оценивают исключительно хорошие товары и т. д. Поэтому среднее отклонение, пользователей учитывается для нормализации вклада оценок таких пользователей. Нормализация была предложена в [23]:

$$p_{ai} = \overline{v_a} + \sigma_a \frac{\sum_{u \in Neigh_a} \left[\left(\frac{v_{ui} - \overline{v_u}}{\sigma_u} \right) s(a,u) \right]}{\sum_{u \in Neigh_a} s(a,u)}$$

1.2 Item-based

Item-based алгоритмы похожи на user-based, но вместо поиска соседей среди пользователей, он ищет похожие предметы. Как и user-based алгоритмы,

могут использоваться различные стратегии для расчета меру сходства. В работе [] предложен скорректированный метод косинусной меры схожести, который согласно автору, дает наилучшие результаты:

$$s(i,j) = \frac{\sum_{u \in U}(v_{ui} - \overline{v_u})(v_{uj} - \overline{v_u})}{\sqrt{\sum_{u \in U}(v_{ui} - \overline{v_u})^2 \sum_{u \in U}(v_{uj} - \overline{v_u})^2}}$$

После того, как будут выбраны N соседей с наилучшей косинусной мерой сходства, для получения оценок на их основе можно пересчитать рекомендации для выбранного объекта.

$$p_{aj} = \frac{\sum_i (s(j,i) v_{ai})}{\sum_i |s(j,i)|}$$

Одним из преимуществ этого алгоритма над user-based решением является то, что сходство между элементами имеет тенденцию быть более постоянным, чем сходство между пользователями, поэтому их можно просчитать заранее.

1.3 Similarity Fusion

Одной из проблем, связанной с предыдущими подходами, является тот факт, что в них используются лишь часть информации, представленной в матрице оценок: либо отношение между пользователями, либо между объектами. Учитывая то, что матрицы оценок обычно разрежены в большинстве реальных систем (обычно пользователи оценивают небольшое число пунктов), было бы логично использовать как можно больше информации. [24] предложил альтернативный метод, который сочетает оба базовых подхода.

Для вычисления предсказаний алгоритм использует рейтинги пользователей, близких к активному пользователю(SUR), рейтинги активного пользователя о близких объектах(SIR), и наконец, рейтинги соседей-пользователей о соседях-объектах(SUIR). Как видно из формулы каждый из

компонентов вносится с определенным весом, который зависит от параметров δ и λ:

$$p_{ui} = \sum_{r \in R} P(v_{ui} = r | SUR, SIR, SUIR) = \left(\sum_{r \in R} P(u_{ui} = r | SUIR) \delta \right) + \left(\sum_{r \in R} P(v_{ui} = r | SUR) \delta (1 - \lambda) \right) + \left(\sum_{r \in R} P(v_{ui} = r | SIR)(1 - \delta)(1 - \lambda) \right)$$

1.4 Regression-Based

Это item-based алгоритм, при котором отношение между двумя объектами можно представить как линейную функцию:

$$f_{i,j}(x) = x\alpha_{ij} + \beta_{ij},$$

где параметры α_{ij} и β_{ij} оцениваются при помощи линейной регрессии. Чтобы получить итоговую оценку необходимо скомбинировать оценки, полученные из каждого предиктора. Более подробное описание методов, основанных на регрессии приводится в [25].

1.5 Slope One

Slope one алгоритм основан на предикторах формы $f(x) = x + b$, который выглядит проще, чем те, которые используются в предыдущем подходе. При этом константа b определяется как средняя разность между каждым объектом и объектом для прогнозирования, среди пользователей, которые оценили оба элемента. Окончательная оценка рассчитывается следующим образом:

$$p_{ui} = \overline{v_u} + \frac{1}{|R_j|} \sum_{j \in R_j} \sum_{x \in S_{ji}} \frac{v_{xi} - v_{xj}}{|S_{ji}|}$$

1.6 ALS (Alternating Least Squares)

Этот алгоритм, предложенный в [26], основан на сокращении размерности исходной рейтинговой матрицы. Полученные в ходе разложения новые матрицы позволяют получить скрытые в оценочной матрице атрибуты, позволяющие находить отношения между элементами и устраняя проблемы, вызванные разреженностью матрицы или аномальными оценками.

§2 Метрики точности

2.1 Точность предсказаний

Данная метрика позволяет оценить разницу между предсказанной оценкой системы и реальной оценкой Самой популярной метрикой этого типа является средняя абсолютная ошибка (MAE). Кроме того, используются другие связанные метрики, такие как среднеквадратичная ошибка (MSE), корень из среднеквадратичной ошибки (RMSE) или нормализованная средняя абсолютная ошибка. RMSE стала чрезвычайно популярной в последние годы после использования в конкурсе Netflix Prize.

Mean absolute error (MAE)

Среднюю абсолютную ошибку можно вычислить как абсолютную разность между предсказанием алгоритма и реальными оценками:

$$|\bar{E}| = \frac{\sum_i^N |p_i - v_i|}{N}$$

Несмотря на некоторые ограничения при оценке систем, ориентированных на рекомендации определенного количества объектов, простота вычисления и статистические свойства сделали эту метрику одной из самых популярных при оценке рекомендательных систем.

Root mean squared error (RMSE)

Полученная с помощью прошлой метрики среднеквадратическая ошибка, рассчитанная с использованием следующей формулы, вносит больший вклад при больших ошибках в предсказаниях:

$$|\bar{E}| = \sqrt{\frac{\sum_i^N (p_i - v_i)^2}{N}}$$

Основная причина использования этой метрики заключается в том, что ошибки, выявленные с помощью этой метрики, могут оказать более сильное влияние на решение пользователя.

2.2 Точность классификации

Метрики данного типа помогают узнать, насколько хорошо система отличает хорошие объекты от плохих. Примерами хорошо известных метрик этого типа являются полнота, точность и ROC-показатели. Эти показатели подходят для задачи поиска самых релевантных объектов, особенно когда предпочтения пользователей выражаются бинарными оценками. Напротив, если пользователи выставляют свои оценки в широком числовом диапазоне, эти показатели не позволяют оценивать правильный порядок пунктов в списке рекомендаций. Эти метрики лишь позволяют узнать хороши или нет рекомендуемые объекты, не учитывая, какой предмет лучше. В таких случаях эта метрика не является лучшим решением.

2.3 Точность ранжирования

С помощью подобных метрик можно узнать насколько точно система может ранжировать выдачу рекомендуемых элементов.

Точность и полнота

Точность можно определить как отношение релевантных объектов ко всем рекомендуемым объектам. Полнота - доля релевантных объектов, которые были рекомендованы пользователю из общего числа релевантных объектов. Понятно, что для системы желательно иметь высокие значения полноты и точности рекомендации. Однако обе метрики связаны следующим образом: когда точность увеличивается, отзыв обычно уменьшается, и наоборот. Тем не менее метрики можно связать через меру F1:

$$F1 = \frac{2PR}{P + R}$$

ROC-кривая

Данная метрика обычно применяется в качестве альтернативы точности/полноте, и часто используется в теории сигналов. Она позволяет наглядно показать поведение системы при классификации релевантных и нерелевантных объектов. Кривая ROC представляет собой графическую интерпретацию верной и ошибочной классификацией объектов.

Глава 3. Реализация рекомендательной системы

§1 Реализация системы

Постановка задачи

На основе входных данных (табл. 4), представляющих собой историю прослушиваний пользователя, сделать предположение о том, какие композиции могли бы его заинтересовать в будущем и выдать список рекомендаций, релевантных для слушателя.

Для этого выполнения поставленной задачи необходимо:

- Реализовать базовые алгоритмы;
- Оценить полученные результаты с помощью выбранных метрик;
- Применить гибридный подход для комбинирования базовых подходов для улучшения результатов

Пользователь	Композиция	Число прослушиваний

Табл. 4. Структура входных данных

Входные данные

В качестве входных данных были использованы данные Million Song Dataset. Этот набор данных содержит:

- Обучающую выборку - полную историю прослушиваний для ~1 миллиона пользователей.
- Набор данных для тестирования - половина истории прослушиваний для 110 тысяч пользователей

После отсеивания избыточных данных, выборка приходит к виду, состоящему из триплетов (табл. 5).

ID пользователя	ID дорожки	Количество прослушиваний
b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOBYHAJ12A6701BF1D	1
b80344d063b5ccb3212f76538f3d9e43d87dca9e	SOCNMUH12A6D4F6E6D	1
b80344d063b5ccb3212f76538f3d9e43d87dca9e	SODACBL12A8C13C273	1
b80344d063b5ccb3212f76538f3d9e43d87dca9e	SODDNQT12A6D4F5F7E	5

Табл. 5. Пример исходных данных

Статистические показатели по обучающим данным, содержащие минимальное, максимальное, среднее и медианное количество пользователей для одной композиции и количество композиций для одного пользователя представлена ниже (табл. 6).

	Минимум	Максимум	Среднее	Медиана
Пользователи /композиция	1	110479	125.794	13
Композиции/ пользователь	10	4400	47.45681	27

Табл. 6. Статистические показатели по обучающим данным

Большая часть композиций была прослушана малым числом пользователей (меньше, чем 13 пользователей для половины всех композиций), а большинство пользователей, в свою очередь, прослушали малое число композиций (меньше чем 27 прослушиваний для половины пользователей).

Apache Spark

Практические эксперименты были проведены на платформе Apache Spark. Apache Spark представляет собой кластерную вычислительную систему с

открытым исходным кодом, направленную на быструю обработку и запись больших объемов данных. Spark может превосходить Hadoop по скорости в 10х раз в итеративных задачах машинного обучения и выполнять запрос, оперируя десятками гигабайт данных с менее секундным временем отклика [27].

В Spark используется Resilient Distributed Dataset (RDD). RDD – это устойчивый распределенный контейнер данных. RDD может быть кэширован в память, а также его можно считывать непосредственно из памяти для процесса последовательной обработки, устраняя необходимость в большом количестве обращений к диску. Это особенно полезно для итеративного вычисления в алгоритмах матричной факторизации.

User-based и item-based алгоритмы

Как уже было отмечено, алгоритмы, реализующие user-based и item-based подходы имеют одинаковую реализацию, основой которой является подсчет меры схожести между слушателями или композициями.

Так как матрица прослушиваний достаточно разреженная, предпочтительнее использовать косинусную меру схожести, вместо других подходов, например, корреляции Пирсона.

В случае User-based подхода:

$$w(u_x, u_y) = \frac{\sum_{i \in S} X_i Y_i}{\sqrt{\sum_{i \in S} X_i} \sqrt{\sum_{i \in S} Y_i}}$$

где S - множество всех композиций, X и Y - история прослушиваний i-го пользователя. Тогда оценку для композиции s_k для пользователя a_k можно рассчитать как $r_{u_a, s_k} = \sum_{i \neq a} I(u_i, s_k) w(u_a, u_{i)}$, где $I(u_i, s_k)$, говорит о том, прослушивал ли пользователь i песню k.

Тогда аналогично для item-based подхода:

$$w(s_x, s_y) = \frac{\sum_{i \in U} X_i Y_i}{\sqrt{\sum_{i \in U} X_i} \sqrt{\sum_{i \in U} Y_i}}$$

где U - множество слушателей, вектора X и Y показывают то, какими пользователями была прослушана композиция, а конечная оценка рассчитывается как $r_{u_a, s_k} = \sum_{i \neq k} I(u_a, s_i) w(s_i, s_k)$, где $I(u_a, s_i)$ говорит о том, прослушивал ли пользователь u_a композицию s_i.

Мы можем улучшить данный подход используя обобщенную метрику косинусной схожести:

$$w(u_y \mid u_x) = \frac{\sum_{i \in S} X_i Y_i}{(\sum_{i \in S} X_i)^q (\sum_{i \in S} Y_i)^{1-q}}$$

Подобным образом, меняя параметр, можно экспериментировать с полученным результатом: при уменьшении значения параметра q, будет увеличиваться вклад, вносимый пользователем u_y.

Гибридный алгоритм

Основываясь на полученных с помощью двух базовых подходов матрицах оценок, можно попытаться улучшить результат, применив гибридную методику. Тогда конечная модель выглядит как:

$$R_{u_a}(s_k) = (1 - \alpha) * UB(R_{u_a}(s_k)) + \alpha * IB\big(R_{u_a}(s_k)\big)$$

ALS-алгоритм

ALS принимает набор обучающих данных и несколько параметров, которые будут использоваться для создания модели. Для определения наилучших значений параметров, будут обучены несколько моделей, а затем будет выбрана лучшая модель, параметры которой будут использоваться для получения результатов.

Процесс поиска лучшей модели состоит из следующих шагов:

1. Выбрать параметры модели; наиболее важный параметр - ранг r (число скрытых факторов). Выбор более низкого ранга ведет к получению больших ошибок, но высокий ранг может привести к переобучению. Вторым параметром будет выступать параметр регуляризации λ.
2. Для каждого набора параметров построим модель и обучим ее с помощью обучающей выборки.
3. Для каждой модели вычислим метрики на основе проверочной выборки и выберем ту, которая покажет наилучшую оценку. Подбор лучших параметров осуществляется в табл. 6.

r	4	8	12	4	8	12	4	8	12
λ	0.1	0.1	0.1	0.2	0.2	0.2	0.3	0.3	0.3
RMSE	5.887	5.432	4.980	5.621	5.221	4.621	5.321	4.781	4.644

Табл. 6. Подбор наилучших параметров модели

Таким образом, наименьшую ошибку показала модель, с параметрами r = 12, λ = 0.2.

§2 Оценка качества

Поскольку в качестве оценок в исходной матрице использовалось количество прослушиваний - как, правило неотрицательное и неограниченное значение, использование метрик, говорящих о точности предсказаний, будет неправильным. Это связано с тем, что перед нами не стоит задачи оценить вероятное число прослушиваний конкретного трека конкретным юзером, а лишь предложить наиболее вероятные рекомендации. Поэтому для задачи в области

музыкальных можно воспользоваться оценками, связанными с точностью выдачи рекомендаций: полнота, точность, F1-мера и тд.

Для задачи были получены следующие значения F-меры(табл. 7, табл. 8, табл. 9).

q	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
F-мера	0,5283	0,5294	0,5314	0,5302	0,5282	0,5211	0,5163	0,512	0,5103	0,5084	0,5051

Табл.7. Зависимость метрик от коэффициента q, user-based

q	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
F-мера	0,5346	0,5381	0,5411	0,5443	0,5432	0,5401	0,5388	0,5362	0,5327	0,5289	0,5231

Табл. 8. Зависимость F-меры от коэффициента q, item-based

α	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9
F-мера	0,5381	0,5391	0,5403	0,5417	0,5424	0,5453	0,5478	0,5484	0,5498

Табл. 9. Зависимость метрик от коэффициента взвешивания α, hybrid

На рис.4 представлены зависимость F1-меры от коэффициента q.

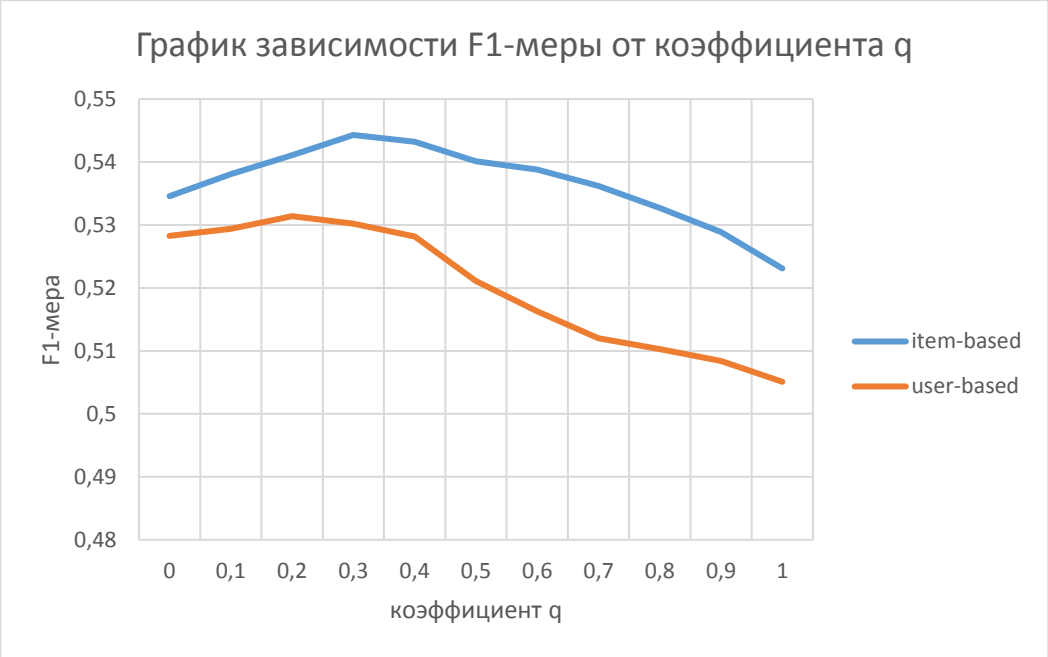

Рис.4. График зависимости F1-меры от коэффициента q

Итоговые результаты представлены в таблице 10.

Алгоритм	F1-мера
User-based коллаборативная фильтрация	0.5314
Item-based коллаборативная фильтрация	0.5443
Гибридная модель	0.5498
ALS	0.5537

Табл. 10. Итоговые результаты

§3 Выводы

Результатом, полученным в данной главе, является программный компонент, решающий задачу генерации музыкальных рекомендаций. Для построения рекомендательной системы реализовано несколько алгоритмов коллаборативной фильтрации: user-based и item-based подходы, их гибридная взвешенная модель, Alternating Least Squares.

Для оценки качества работы была проведена оценка точности на основе F-меры. Для user-based, item-based и гибридного взвешенного алгоритмов были найдены значения параметров, при которых эти подходы дают оптимальный результат. Наилучшие оценки соответственно: 0.5314 - для user-based алгоритма, 0.5443 - для item-based, 0.5498 - для гибридного взвешенного подхода, 0.5537 - для ALS. Можно заметить, что item-based алгоритм дает более качественную оценку, чем user-based, а при построении линейной комбинации этих моделей

можно получить несколько улучшенный результат, что и предполагается при построении гибридных моделей.

Более высокие результаты алгоритма, основанного на факторизации, могут объясняться тем, что традиционные алгоритмы коллаборативной фильтрации не могут давать столь эффективный результат из-за сильной разреженности исходных данных.

Заключение

В ходе данной работы были рассмотрены основные виды рекомендательных систем и принципы их построения. Был приведен подробный обзор алгоритмов коллаборативной фильтрации, показаны способы оценки качества подобных подходов.

В качестве базового подхода к решению задачи был выбран метод, основанный на коллаборативной фильтрации и реализованы алгоритмы, основанные как на близости пользователей и объектов, так и на факторизации исходных данных. Лучший результат показал алгоритм ALS, этот факт можно объяснить большой разреженностью исходных данных.

Работа может быть продолжена путем улучшения базовых алгоритмов, экспериментов с построением других гибридных моделей, использования дополнительных метаданных для создания систем, основанных на знаниях и систем на основе контента.

Список литературы

1. Hill W., Stead L., Rosenstein M., Furnas G. Recommending and Evaluating Choices in a Virtual Community of Use // Proceeding Conference Human Factors in Computing Systems, 1995. P. 194-201.
2. Resnick P., Iakovou N., Sushak M., Bergstrom P., Riedl J. GroupLens: An Open Architecture for Collaborative Filtering of Netnews // Proceeding 1994 Computer Supported Cooperative Work Conference, 1994. P. 175-186.
3. Shardanand U., Maes P. Social Information Filtering: Algorithms for Automating «Word of Mouth» // Proc. Conf. Human Factors in Computing Systems, 1995 . P. 210-217.
4. Goldberg D., Nichols D., Oki B. M., Terry D. Using collaborative filtering to weave an information Tapestry // Special issue on information filtering, 1992. Vol. 35, Issue 12 P. 61-70.
5. Adomavicius G., Tuzhilin A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions // IEEE Transactions on Knowledge and Data Engineering, 2005. Vol. 17, Issue 6. P 734-749.
6. Celma O. Music Recommendation and Discovery. Springer-Verlag Berlin Heidelberg, 2010. 194 p.
7. Ricci F., Rokach L., Shapira B., Kantor P.B. Recommender Systems Handbook. Springer US, 2011. 842 p.
8. Aggarwal C. C. Data mining. The Textbook. Springer International Publishing, 2015. 734 p.
9. Mango T., Sable C. A comparison of signal-based music recommendation to genre labels, collaborative filtering, musicological analysis, human recommendation,

and random baseline // Proceedings of the 9th international conference on music information retrieval, 2008. P. 161-166.

10. Knees P., Pohle T., Schedl M., Widmer G. Combining audio-based similarity with web-based data to accelerate automatic music playlist generation // Proceedings of the 8th ACM international workshop on Multimedia information retrieval, 2006. P. 147-154.

11. Schedl M., Flexer A., Urbano J. The neglected user in music information retrieval research // Journal of Intelligent Information Systems, 2013. Vol. 41, Issue 3. P. 523-539.

12. Gabriel H. H., Spiliopoulou M., Nanopoulos A. Eigenvectop-based clustering using aggregated similarity matrices // Proceedings of the 2010 ACM Symposium on Applied Computing, 2010. P. 1083-1087.

13. Королева Д. Е., Филиппов М. В. Анализ алгоритмов обучения коллаборативных рекомендательных систем // Инженерный журнал: наука и инновации, 2013. Вып. 6. Стр. 1-8.

14. Николенко С.А. Рекомендательные системы. СПб: Изд-во Центр Речевых Технологий, 2012. 53 с.

15. Berry M.W. Large scale singular value computations // International Journal of Supercomputer Applications, 1992. No. 6(1). P. 13–49.

16. Billsus D., Pazzani M.J. Learning Collaborative Information Filters // Proceeding 15th International Conference on Machine Learning, 1998. P. 46-54.

17. Ekstrand M. D., Riedl J. T., Konstan J. A. Collaborative Filtering Recommender Systems // Foundations and Trends® in Human–Computer Interaction, 2011. Vol. 4, No. 2. P. 81-173.

18. Goldberg K., Roeder T., Gupta D., Perkins C. Eigentaste: A constant time collaborative filtering algorithm // Information Retrieval, 2001. Vol. 4, No. 2. P. 133–151.

19. Aggarwal C. C. Recommender Systems. The Textbook. Springer International Publishing, 2016. 498 p.

20. Джонс M. Рекомендательные системы. https://www.ibm.com/developerworks/ru/library/os-recommender1/index.html

21. Pazzani M., Billsus D. Learning and revising user profiles: The identification of interesting web sites // Machine Learning - Special issue on multistrategy learning, 1997. Vol. 27, Isue 3. P. 313–331.

22. Jannach D., Zanker M., Felfernig A., Friedrich G. Recommender Systems – An Introduction. Cambridge University Press, 2010. 360. P

23. Herlocker J., Webster J., Jung S., Dragunov A., Holt T., Culter T., Haerer S. A framework for collaborative information environments and unified access to distributed digital content // Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries, 2002. P. 378-378.

24. Wang A. The Shazam music recognition service // Communications of the ACM, 2006. P. 44-48.

25. Vucetic S., Obradovic Z. Performance Controlled Data Reduction for Knowledge Discovery in Distributed Databases // Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications, 2000. P. 29-39.

26. Sarwar B., Karypis G., Konstan J., Riedl J. Analysis of recommendation algorithms for e-commerce // Proceedings of the 2nd ACM conference on Electronic commerce, 2000. P. 158-167.

27. Zaharia M., Chowdhury M., Franklin M. J., Shenker S., Stoica I. Spark: cluster computing with working sets // HotCloud, 2010. Vol. 10, P. 10.