Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования «Южно-Уральский государственный университет
(национальный исследовательский университет)»
Факультет математики, механики и компьютерных технологий
Кафедра прикладной математики и программирования
Направление подготовки Прикладная математика и информатика

РАБОТА ПРОВЕРЕНА
Рецензент,
_____
«____»_____2017г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой, д.ф.–м.н.,
доцент
_____/А.А.Замышляева
«____»_____2017 г.

Разработка и исследование алгоритма коллаборативной фильтрации с учетом временного фактора

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
ЮУрГУ–01.03.02.2017.228.ПЗ ВКР

Руководитель работы, доцент
_____/Т.Ю.Оленчикова
«____»_____2017 г.
Автор работы
Студентка группы ЕТ–482
_____/Р.Д.Байтурина
«____»_____2017 г.
Нормоконтролер, доцент
_____/Т.Ю.Оленчикова
«____»_____2017 г.

Челябинск 2017

АННОТАЦИЯ

Байтурина Р.Д. Разработка и исследование алгоритма коллаборативной фильтрации с учетом временного фактора.– Челябинск: ЮУрГУ, ЕТ–482, 33 с., 4 ил., 5 табл., библиогр. список – 24 наим., 1 прил.

Выпускная квалификационная работа посвящена разработке и исследованию алгоритма коллаборативной фильтрации, учитывающего временной фактор. В работе рассмотрены основные алгоритмы для рекомендательных систем и проанализированы существующие методы коллаборативной фильтрации. Разработан алгоритм прогнозирования рекомендаций с учетом временного фактора. Описана реализация программы на языке T-SQL для расчёта оценки для объектов из базы данных MovieLens. Показаны графики погрешностей прогнозируемого рейтинга.

Оглавление

ВВЕДЕНИЕ

В современном мире мы часто можем столкнуться с проблемой рекомендации товаров и услуг пользователям какого-либо сайта, информационной системы. Раньше для формирования рекомендаций вполне хватало сводки наиболее популярных продуктов: это используется и сейчас. Но постепенно с течением времени такие рекомендации стали вытесняться целевыми предложениями: пользователям предлагаются именно те продукты, которые почти наверняка понравятся именно им.

Одним из методов построения прогнозов в рекомендательных системах, который использует реакцию одного пользователя на объекты, которая обычно представлена в виде оценок, для прогнозирования неизвестных предпочтений другого пользователя, является метод коллаборативной фильтрации.

Основная идея данного метода заключается в том, что те люди, которые давали одинаковую оценку каким-либо объектам или товарам в прошлом, склонны давать похожие оценки другим предметам и в будущем. Прогнозы составляются индивидуально для каждого пользователя, хоть и используемая информация собрана от многих участников. Этим коллаборативная фильтрация отличается от более простого подхода, который дает среднюю оценку для каждого объекта, к примеру, основывающуюся на количестве отданных за него голосов, или на популярности.

В данной работе мы рассмотрим метод коллаборативной фильтрации, как самый распространенный. Его можно модифицировать, получая наиболее хорошие результаты. Интересы людей со временем могут меняться, поэтому классические алгоритмы коллаборативной фильтрации становятся менее актуальными в условиях увеличивающегося количества пользователей, объемов интернет-контента, большей активности пользователей и изменчивости их предпочтений. Поэтому в данной работе мы рассматриваем модифицированный алгоритм коллаборативной фильтрации, который учитывает временной фактор, и сравниваем его с тем же методом, который время не учитывает.

1 ОБЗОР СУЩЕСТВУЮЩИХ МЕТОДОВ ПРОГНОЗИРОВАНИЯ НА ОСНОВЕ КОЛЛАБОРАТИВНОЙ ФИЛЬТРАЦИИ

1.1 Базовые алгоритмы коллаборативной фильтрации

Рекомендательные системы — программы, которые пытаются предсказать, какие объекты (музыка, новости, фильмы, книги, сайты) могут понравиться пользователю, имея некоторую определенную информацию о его профиле.

Два основных подхода для создания рекомендательных систем — коллаборативная фильтрации и фильтрация на основе содержания. При фильтрации на основе содержания создаются профили юзеров и объектов, профили пользователей могут включать информацию о возрасте, гендерной принадлежности, роде деятельности или ответы на вопросы определенной анкеты, профили объектов могут включать атрибутивную информацию в зависимости от типа объекта: названия фильмов, имена актеров, исполнителей, названия жанров. В процессе работы рекомендательные системы явно или неявно собирают данные о пользователях. Например, явный сбор данных может включать:

- запрос у пользователя оценки какого-либо объекта;
- запрос ранжирования группы объектов от наиболее понравившегося к наименее интересному для конкретного пользователя;
- дать пользователю два объекта и попросить выбрать наиболее интересный;
- предложение создать список объектов, любимых пользователем.

Неявный сбора данных может проявляться так:

- отслеживание, какие товары рассматривает пользователь, например, в интернет-магазинах;
- фиксирование поведения пользователя онлайн.

Рекомендательные системы сравнивают однотипные данные от разных пользователей и вычисляют примерный список рекомендаций для конкретного пользователя. Для вычисления рекомендаций удобно использовать граф интересов. Рекомендательные системы — это альтернатива для поисковых алгоритмов, так как позволяют найти объекты, которые не могут быть найдены последними.

В данной работе будет рассматриваться метод коллаборативной фильтрации для рекомендательных систем.

Главная идея алгоритмов коллаборативной фильтрации заключается в предложении новых объектов для конкретного пользователя на основе предыдущих предпочтений пользователя или мнения других единомышленников данного пользователя. На сегодняшний день исследователи разработали целый ряд алгоритмов коллаборативной фильтрации, которые можно разделить на три основные категории [1].

1. Методы, основанные на анализе имеющихся оценок, – анамнестические методы (Memory-based). Эти алгоритмы основываются на статистических методах, чтобы найти группу пользователей близких к целевому пользователю. Этот подход также называют методом ближайших соседей: использование предшествующих оценок, сделанных клиентом, и анализ оценок других пользователей, которые имеют подобные предпочтения. Тогда прогноз для целевого пользователя формируется на основании вычисления некой меры схожести по всем накопленным данным.

2. Методы, основанные на анализе модели данных, – модельные методы (Model-based). В этом случае сначала по совокупности оценок формируется описательная модель предпочтений пользователей, товаров и взаимосвязи между ними, а затем формируются рекомендации на основании полученной модели. Процесс формирования рекомендаций разбит на два этапа: ресурсоемкое обучение модели в отложенном режиме и достаточно простое вычисление рекомендаций на основе существующей модели в реальном времени. Эти алгоритмы могут быть основаны на вероятностном подходе, кластерном анализе, анализе скрытых факторов.

3. Гибридные методы – объединяют коллаборативную фильтрацию с другими технологиями рекомендаций (обычно с системами, анализирующими содержимое) для прогнозирования. Системы, анализирующие содержимое дают рекомендации на основе текстовой информации, такой как документы, ссылки, новостные сообщения, веб-журналы, описания объектов и профилей пользователей об их вкусах, потребностях, предпочтениях и находят закономерности в содержании. Множество элементов способствует важности текстового содержания, такие как наблюдаемые особенности просмотренных слов или страниц и сходство между объектами, которые понравились пользователю в прошлом. Затем рекомендательные системы, анализирующие содержимое используют эвристические методы или алгоритмы классификации для выработки рекомендаций. Технологии на основе фильтрации по содержимому имеют стартовую проблему, в которой они должны иметь достаточно информации, чтобы построить надежный классификатор. Кроме того, они ограничены признаками, явным образом связанными с объектами, которые они рекомендуют (иногда эти признаки трудно извлечь), в то время как коллаборативная фильтрация может давать рекомендации без каких-либо описательных данных. Так же эти технологии имеют проблемы сверхспециализации, то есть они могут рекомендовать только объекты, которые высоко оценены в профиле пользователя или в его/ее истории оценок. (Рисунок 1.1)

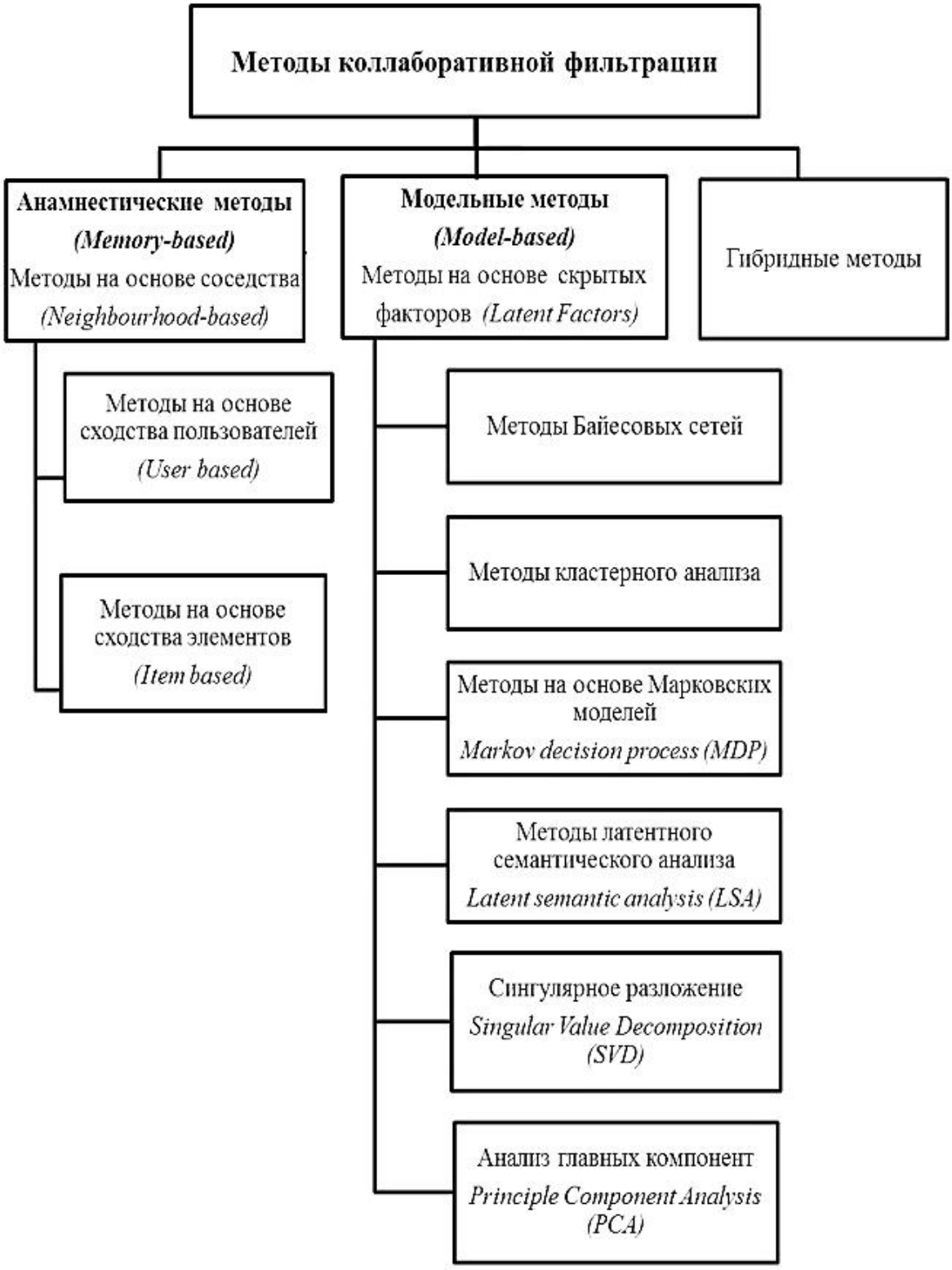

Рисунок 1.1 – Методы коллаборативной фильтрации

Алгоритмы на основе соседства подходят для совместной фильтрации, используя полную базу данных. Как описали в статье Billsus D., Pazzani M. [13], алгоритм пытается найти пользователей, которые похожи на активных пользователей (то есть пользователей, оценки которых мы хотим предсказать), и

использует их предпочтения для прогнозирования оценок для активного пользователя.

Другие рекомендательные системы включают в себя рекомендательные системы, основанные на демографии, которые используют информацию пользовательского профиля, такую как пол, почтовый индекс, род занятий и т.д.; рекомендательные системы, основанные на полезности и рекомендательные системы, основанные на знаниях, требуют знания о том, как конкретный объект удовлетворяет потребности пользователей.

В надежде избежать ограничений любой рекомендательной системы и повысить производительность рекомендации, гибридные рекомендательные системы коллаборативной фильтрации объединены путем добавления характеристик на основе содержимого в модели коллаборативной фильтрации, объединяют коллаборативную фильтрацию с анализирующей содержимое или другими системами или объединяют разные алгоритмы коллаборативной фильтрации.

Анамнестические методы в свою очередь разделяются на:

- анализ сходства пользователей (User-based);
- анализ сходства элементов (Item-based).

Целью этих двух направлений является объединение похожих объектов в группы на основе матрицы оценок.

В первом случае определяется сходство пользователей: найти других пользователей, чьи прошлые оценки похожи на те, что у текущего пользователя, и использовать их оценки других элементов для прогнозирования предпочтений текущего пользователя.

В случае второго подхода вместо того, чтобы использовать подобие между поведением пользовательских оценок для прогнозирования предпочтения, используется сходство между оценками моделей элементов. Если два элемента, как правило, имеют одинаковые оценки пользователей, то они похожи, и пользователи должны иметь аналогичные предпочтения для подобных элементов.

У метода User-Based есть достаточно весомый недостаток - при увеличении числа пользователей сложность вычисления персональной рекомендации линейно увеличивается. То есть использование этого метода будет оправдано только в том случае, когда количество объектов тоже достаточно большое, иначе рекомендуется использовать метод Item-based.

Для определения сходства между пользователями или элементами можно использовать такие подходы:

- расстояние Эвклида,
- корреляция Пирсона,
- ранговая корреляция Спирмена,
- коэффициент Жаккара,
- косинусное подобие.

Чтобы измерить сходство, можно найти корреляцию между двумя пользователями. Это даст значение от -1 до 1, которое определяет, насколько

подобны два пользователя. Значение 1 означает, что оба они оценивают объект одинаковым образом, тогда как значение -1 означает, что они оценивают объекты точно противоположно (то есть один высоко, другой низко или наоборот).

Метод на основе схожести элементов

Шаг 1: для каждого элемента j находится его мера близости к элементу i. Для этого можно воспользоваться один из подходов, про которые говорилось выше, например, коэффициент корреляции Пирсона[1]:

$$¿(i,j)=\frac{\sum_u (r_{u,i}-\bar{r}_u)*(r_{u,j}-\bar{r}_u)}{\sqrt{\sum_u (r_{u,i}-\bar{r}_u)^2}*\sqrt{\sum_u (r_{u,j}-\bar{r}_u)^2}}$$

Шаг 2: выбираем множество элементов K, наиболее похожих на объект i.

Шаг 3: предсказание оценки (рейтинга) объекта на основе оценок похожих на него объектов[1]:

$$\widehat{r_{u,i}}=\frac{\sum_j ¿(i,j)*r_{u,j}}{\sum_j ¿(i,j)}$$

В методах группы user-based сходство ищется между пользователями и в качестве рекомендаций пользователю выдается k самых часто покупаемых товаров n наиболее похожи на него покупателями. Практика показывает, что удобнее брать top-K ближайших соседей.

Топ-K рекомендаций – это набор K топ-оцененных объектов, которые будут интересны для какого-то определенного пользователя. Например, если Вы зарегистрированный клиент какого-либо сайта, использующего систему коллаборативной фильтрации, и где предусмотрена такая функция, после входа в свою учетную запись, можете порекомендовать список фильмов/книг (или других продуктов), которые могут представлять ваш интерес. Техника топ-K рекомендаций анализирует матрицу юзер-объект, чтобы обнаружить отношения между различными пользователями или объектами и использовать их для вычисления рекомендаций. Некоторые модели, такие как модели, основанные на поиске ассоциативных правил, могут быть использованы для построения топ-K рекомендаций.

А) Алгоритмы топ-K рекомендаций по пользователю.

«Эти алгоритмы сначала определяют k наиболее схожих пользователей (ближайших соседей) для активного пользователя, используя корреляцию Пирсона или модель векторного пространства, в которой каждый пользователь рассматривается как вектор в m-размерном пространстве и схожести между активным пользователем и другими пользователями вычисляется между векторами. После нахождения k наиболее похожих пользователей, их соответствующие строки в матрице R пользователь-объект объединятся для определения набора объектов, приобретенного группой вместе с их частотой. После получения набора метод коллаборативной фильтрации по пользователю рекомендует топ K наиболее частых объектов в выбранном наборе, которые активный пользователь еще не приобретал. Алгоритм рекомендаций топ K по

пользователю имеет ограничения, связанные с масштабируемостью и производительность в реальном времени»[4].

Определить множество ближайших соседей (соседство) для целевого пользователя можно по формуле:

$$N(u_0)=\{u \mid ¿(u_0,u)\geq\theta\}$$

После упорядочения пользователей по убыванию сходства, нужно отбирать не только первые k, но и проверить значение сходства у следующего (k+1) по списку. Если это значение совпадает с последним, включать и k+1-го пользователя в соседство. И так до тех пор, пока сходство не изменится. Поскольку нужно предсказать оценку целевым пользователем конкретного предмета i, то интересны только те пользователи из этого соседства, которые оценивали предмет i:

$$N(u_{0,i})=\{u_{0,i} \mid i\in u',u\in N(u_0)\}$$

Пусть $\widehat{r_{u,i}}$ - оценка предмета i пользователем u, таким образом, итоговая формула имеет вид:

$$\widehat{r_{u,i}}=\frac{\sum_u r_{u,i}\times\sim(u_0,u)}{\sum_u ¿(u_0,u)}$$

Чтобы выдать пользователю рекомендацию, нужно посчитать такие предсказания для каждого из неоцененных им предметов, ранжировать по убыванию и выдать первые k предметов в качестве рекомендованных.

Б) Алгоритмы топ-K рекомендаций по объекту.

«Эти алгоритмы были разработаны для решения проблемы масштабируемости алгоритмов топ-K рекомендаций по пользователю. Они сначала вычисляют k наиболее похожих объектов для каждого объекта соответствующего схожести; затем определяют набор C, как кандидатов рекомендуемых объектов, беря объединение наиболее похожих k объектов и удаления каждого из объектов в наборе U, что пользователь уже приобрел. Результирующий набор объектов в C, отсортированный в порядке уменьшения сходства, будет рекомендованным списком. Одна из проблем этого метода состоит в том, что когда совместно распределенный набор объектов отличается от распределения отдельных объектов в наборе, вышеуказанные схемы потенциально могут давать не оптимальные рекомендации. Для решения этой проблемы Deshpande и Karypis разработали алгоритмы топ-K рекомендаций по предмету более высокого порядка, он использует все комбинации объектов до определенного размера при определении наборов объектов, которые будут рекомендованы пользователю»[4].

Вводятся обозначения: u_0 – «целевой» пользователь, u_{0l} – предметы, которые он оценивал, sim(i,j) – сходство предмета i с предметом j. Соседство для предмета i определяется аналогично тому, как определяется оно для целевого пользователя:

$$N(i)=\{u \mid ¿(i,j)\geq\theta\}$$

Чтобы предсказать оценку целевому пользователю, нужно сравнивать те предметы, которые он оценивал, с теми, которые нет [$1]. Поэтому формула для соседства в отношении к целевому пользователю выглядит так:

$$N(i\vee u_0)=\{j \mid j\notin u',i\in u',j\in N(i)\}$$

Пусть $\widehat{r_{u,i}}$ - оценка предмета i пользователем u, таким образом, итоговая формула имеет вид:

$$\widehat{r_{u,i}} = \frac{\sum_j r_{u,i} \times \sim(j,i)}{\sum_j ¿(j,i)}$$

Далее нужно ранжировать оценки по убыванию. Выдается первые k в качестве рекомендаций. Преимущества этому методу дает тот факт, что обычно на сайтах онлайновой торговли количество пользователей постоянно увеличивается, в то время как новые предметы добавляются не так часто. Поэтому вычисление сходства пользователей каждый раз при формировании рекомендации может занимать много времени, а сходство предметов можно посчитать заранее в режиме офлайн, и потом использовать эту матрицу.

Коллаборативная фильтрация на основе сходства пользователей (User-based) имеет высокую точность. Однако недостатком является ресурсоемкость (требования к памяти) и сложность (количество вычислений, которое требуется для получения рекомендаций). К тому же вычисление степени схожести может производиться только в реальном времени, так как данные о текущей транзакции становятся доступными только в момент выработки рекомендаций. Поэтому данный метод может применяться только к относительно небольшим базам данных. В алгоритме на основе сходства элементов (Item-based) степень близости анализируемого элемента ко всем остальным может быть вычислена в отложенном режиме по расписанию, так как вектора рейтингов всех элементов доступны до момента формирования рекомендации. Таким образом, этот алгоритм оказывается более эффективным с точки зрения времени формирования прогнозов благодаря возможности проведения отложенной предобработки данных.

1.2 Альтернативные алгоритмы и расширения

Некоторые исследователи наряду с общепринятой классификацией, приведенной в предыдущем подпункте, выделяют также еще несколько типов алгоритмов [1].

1. Голосование по умолчанию.

«Во многих коллаборативных фильтрах парное сходство вычисляется только из оценок на пересеченных объектах, которые оценили оба пользователя. Это не будет надежным, когда в наличии слишком мало голосов, чтобы генерировать значения сходства. Кроме того, при сосредоточении внимания на схожих наборах пересечений, пренебрегается глобальное поведение оценок, отражающееся во всей истории оценок пользователя.

Эмпирически, предполагается, что некоторые значения голосования умолчанию для отсутствующих оценок могут улучшить производительность прогнозирования КФ».

2. Обратная пользовательская частота.

«Идея обратной пользовательской частоты, применяемой в КФ, заключается в том, что универсально понравившиеся объекты, не столь полезны в определении сходства, как менее общие объекты. Обратная частота может быть определена как:

$$f_j = \log \frac{n}{n_j} \quad ,$$

где n_j - число пользователей, которые оценили объект j;
n – общее число пользователей.

Если все оценят объект j, тогда f_j равно 0. Для того, чтобы применить обратную пользовательскую частоту при использовании алгоритма КФ, основанного на векторе сходства, необходимо использовать преобразованную оценку, которая является простой первоначальной оценкой умноженной на коэффициент f_j».

3. Коэффициент усиления.

«Относится к преобразованию, применяемому к весам, используемым в основе предсказаний КФ. Преобразование подчеркивает высокие веса и опускает низкие веса.

Коэффициент усиления снижает уровень шума в данных. Как правило, предпочтительные высокие веса, имеющие маленькие значения, при возведении в степень становятся пренебрежительно малыми».

4. Алгоритмы КФ увеличения-подстановки.

«Когда оценочные данные для задач КФ сильно разрежены, возникает проблема сделать точные предсказания, используя КФ, основанную на корреляции Пирсона, предлагают основу коллаборативной фильтрации увеличения-подстановки (IBCF – imputation-boosted CF), которая впервые использует методы подстановки, чтобы заполнить недостающие данные, перед использованием традиционного алгоритма КФ, основанного на корреляции Пирсона на этих заполненных данных для прогнозирования конкретной пользовательской оценки для указанного объекта».

5. Взвешенное предсказание большинства.

«Этот алгоритм делает свои предсказания, используя строки с данными наблюдения в тех же столбцах, взвешенных путем проверенного сходства между строками, с бинарными значениями оценок.

Прогноз для оценки на определенном объекте для активного пользователя определяется оценкой на объекте определенного пользователя, который имеет самое высокое накопленное значение веса с активным пользователем. Этот алгоритм можно обобщить для данных мультикласса, а так же расширить от сходства пользователь-пользователь до сходства объект-объект и к пользователь-объект комбинированного сходства. Одним из недостатков этого алгоритма является масштабируемость, когда число пользователей или объектов растет выше определенного большого числа n, будет непрактичным для вычислений сходства пользователь-пользователь или объект-объект обновлять матрицу подобия».

1.3 Коллаборативная фильтрация с учетом временного фактора

Интересы и предпочтения пользователей могут меняться с течением временем. Поэтому это необходимо учитывать, чтобы повысить качество рекомендаций. Для решения данной проблемы существует три основных подхода:

1. Выбираются наиболее важные оценки, как правило, за последний определенный отрезок времени, и только выбранные оценки будут использоваться при вычислении рекомендаций.
2. Каждой оценке в соответствие ставится вес. Наиболее поздним оценкам соответствует больший вес.
3. Использование неких моделей (систем предсказателей). Предсказатели взвешиваются согласно релевантности к текущему моменту времени. Например, модели, показавшие наилучший результат на текущих оценках, получают наиболее высокий вес.

1.4 Проблемы коллаборативной фильтрации

Рассмотрим основные проблемы, связанные с коллаборативной фильтрацией[2].

1.4.1 Разреженность данных

«Как правило, большинство коммерческих рекомендательных систем основано на большом количестве данных, в то время как большинство пользователей не ставит оценки товарам. В результаты этого матрица «предмет-пользователь» получается очень большой и разреженной, что представляет проблемы при вычислении рекомендаций. Эта проблема особенно остра для новых, только что появившихся систем. Также разреженность данных усиливает проблему холодного старта».

1.4.2 Масштабируемость

«Когда количество существующих пользователей и объектов чрезвычайно растет, традиционные алгоритмы коллаборативной фильтрации будут страдать от серьезных проблем масштабируемости, с вычислительными ресурсами, выходящими за рамки практических или приемлемых уровней. Например, с десятками миллионов пользователей (M) и миллионов отдельных объектов (N) алгоритм коллаборативной фильтрации со сложностью O(n) уже слишком велик. Кроме того, многие системы должны немедленно реагировать на онлайн требования и давать рекомендации для всех пользователей независимо от их покупательской или рейтинговой истории, которая требует высокой масштабируемости от системы коллаборативной фильтрации».

1.4.3 Проблема холодного старта

«Новые предметы или пользователи представляют большую проблему для рекомендательных систем. Частично проблему помогает решить подход,

основанный на анализе содержимого, так как он полагается не на оценки, а на атрибуты, что помогает включать новые предметы в рекомендации для пользователей. Однако проблему с предоставлением рекомендации для нового пользователя решить сложнее».

1.4.4 Синонимия

«Относится к способности одинаковых или очень похожих объектов иметь одинаковое название или запись. Большинство рекомендательных систем не способны обнаружить эту скрытую ассоциацию, и соответственно рассматривают эти объекты по-разному. Например, казалось бы, разные объекты «детское кино» и «фильм для детей» на самом деле одно и то же, но анамнестическая коллаборативная фильтрация не найдет совпадений между ними, чтобы вычислить сходство. Действительно, степень изменчивости в использовании описательного термина больше, чем обычно подозревают. Распространенность синонимов снижает производительность рекомендации систем коллаборативной фильтрации».

1.4.5 Мошенничество

«В рекомендательных системах, где каждый может ставить оценки, люди могут давать позитивные оценки своим предметам и плохие своим конкурентам. Также, рекомендательные системы стали сильно влиять на продажи и прибыль, с тех пор как получили широкое применение на коммерческих сайтах. Это приводит к тому, что недобросовестные поставщики пытаются мошенническим образом поднимать рейтинг своих продуктов и понижать рейтинг своих конкурентов».

1.4.6 Разнообразие

«Коллаборативная фильтрация изначально призвана увеличить разнообразие, чтобы позволить открывать пользователям новые продукты из бесчисленного множества. Однако некоторые алгоритмы, в частности, основанные на продажах и рейтингах, создают очень сложные условия для продвижения новых и малоизвестных продуктов, так как их замещают популярные продукты, которые давно находятся на рынке. Это в свою очередь только увеличивает эффект «богатые становятся ещё богаче» и приводит к меньшему разнообразию».

1.4.7 Белые вороны

«К «белым воронам» относятся пользователи, чьё мнение постоянно не совпадает с большинством остальных. Из-за их уникального вкуса, им невозможно что-либо рекомендовать. Однако такие люди имеют проблемы с получением рекомендаций и в реальной жизни, поэтому поиски решения данной проблемы в настоящее время не ведутся».

1.5 Базы данных для исследования методов коллаборативной фильтрации

Для исследований методов коллаборативной фильтрации подходят базы данных, содержащие информацию о пользователях, об объектах и об оценках, выставленных этим объектам.

В данной работе в качестве исходных данных для исследований используются данные из базы MovieLens – базы рейтингов фильмов, поскольку эти данные удовлетворяют условиям задачи. Она содержит 3900 фильмов , 1000209 анонимных рейтингов и 6040 пользователей, которые выставляют эти рейтинги. Материалы предоставлены проектом GroupLens. «Исследовательский проект GroupLens является исследовательской группой на факультете компьютерных наук и инженерии в Университете Миннесоты. Члены исследовательского проекта GroupLens Research Project участвуют во многих исследовательских проектах, связанных с такими областями, как фильтрация информации, коллаборативная (или совместная) фильтрация и системы рекомендаций. Проект возглавляют профессора Джон Ридл и Джозеф Констан. В 1992 году проект начал изучать автоматическую совместную фильтрацию, но наиболее известен своей всемирной пробной версией автоматизированной системы совместной фильтрации для новостей Usenet в 1996 году. С тех пор проект расширил сферу своей деятельности, чтобы исследовать общие решения по фильтрации информации, а также улучшение существующей технологии совместной фильтрации»[21].

1.6 Обоснование выбора платформы для исследований

Среда Microsoft SQL Server 2008 Management Studio Express представляет собой интегрированную среду для доступа, настройки, управления, администрирования и разработки всех компонентов SQL Server. Среда SQL Server 2008 Management Studio Express объединяет большое число графических средств с полнофункциональными редакторами сценариев для доступа к SQL Server разработчиков и администраторов с любым уровнем подготовки. Разработчики получают знакомую среду, а администраторы баз данных — единую полнофункциональную программу, объединяющую простые в использовании графические средства и богатые возможности для создания сценариев[20]. Также эта среда является бесплатной, что немаловажно.

Matlab — это пакет прикладных программ для решения задач технических вычислений и одноимённый язык программирования, используемый в этом пакете. Пакет используют более миллиона инженерных и научных работников, он работает на большинстве современных операционных систем, включая, конечно, Microsoft Windows.

Для исследований основным языком программирования был выбран Transact-SQL, так как это было самым удобным вариантом для вычислений. Результаты обрабатывались в Matlab. В нем можно организовать простое построение графиков.

1.7 Постановка задачи

Целью данной работы является разработка алгоритма коллаборативной фильтрации, позволяющего улучшить качество рекомендаций за счет учета временного фактора, а также исследование базовых алгоритмов. В качестве базы данных для исследования выбрана база MovieLens, включающая в себя рейтинги более 3000 фильмов за последние 20 лет, более 1000000 оценок [18]. Учет временного фактора предполагает, что предпочтения пользователей изменяются с течением времени. Задачами работы являются:

1) анализ существующих методов коллаборативной фильтрации;
2) выбор базовых алгоритмов для исследования;
3) разработка на их основе алгоритма с учетом временного фактора;
4) разработка методики экспериментальных исследований;
5) разработка программ и проведение эксперимента;
6) анализ результатов эксперимента.

1.8 Выводы по разделу

В данном разделе выполнен анализ задач, возникающих при разработке рекомендательных систем, а также популярных алгоритмов коллаборативной фильтрации (КФ). Рассмотрены области применения КФ, существующие алгоритмы и проблемы. Коллаборативная или совместная фильтрация – это один из методов построения прогнозов в рекомендательных системах, использующий известные предпочтения (оценки) группы пользователей для прогнозирования неизвестных предпочтений другого пользователя.

Существуют три группы методов коллаборативной фильтрации: методы на основе сходства (анамнестические методы), методы на основе скрытых факторов (модельные методы), гибридные методы.

Достоинствами анамнестических методов являются:

- простота в использовании;
- легко добавлять новые данные, не требуют обучения и переобучения;
- не используют свойства элементов для выдачи рекомендаций;
- хорошая масштабируемость для совместно оценённых элементов.

Из недостатков можно отметить:

- зависимость от рейтингов пользователей;
- качество рекомендаций заметно падает в случае разреженности данных;
- не могут выдать рекомендации новым пользователям (элементам);
- ограниченная масштабируемость для очень больших баз данных.

Модельные методы для предсказания используют различные модели искусственного интеллекта. Они, по сравнению с анамнестическими методами, лучше справляются с разреженными данными и масштабируемостью, обеспечивают лучшее качество предсказаний. Однако эти методы дороги в создании и использовании, требуют обучения, всегда приходится выбирать между масштабируемостью и качеством предсказаний.

Гибридные методы в принципе обеспечивают самое лучшее качество предсказаний, хорошо справляются с проблемами КФ, такими как разреженность данных и др., но они требуют дополнительную информацию о пользователях / элементах , которая не всегда доступна, отличаются повышенной сложностью и стоимостью в создании и обслуживании.

Сформулированы цели и задачи исследований. Для исследований были выбраны анамнестические методы, так как они проще в реализации, хорошо подходят для разработки рекомендательных систем различных сайтов, а модельные и гибридные методы больше подходят для более мощных, профессиональных систем.

Базой данных для исследований была выбрана база рейтингов фильмов MovieLens.

2 РАЗРАБОТКА АЛГОРИТМА КОЛЛАБОРАТИВНОЙ ФИЛЬТРАЦИИ

2.1 Математическая постановка задачи

Рассмотрим в задаче коллаборативной фильтрации метод item-based с учетом временного фактора и без него. Выборка тут представляет собой тройки вида (u; i; $r_{u,i}$), где u – пользователь, i – фильм, $r_{u,i}$– оценка, которую пользователь u поставил фильму i. И будем также считать, что рейтинги нормированы на отрезке [0; 1].

Сначала определим формулы для методов без учета времени.

Метод item-based. Пусть есть матрица пользователь-признак, составленная из оценок пользователей, тогда мера схожести товаров i и j как векторов найдем с помощью коэффициента корреляции Пирсона по формуле:

$$¿(i,j)=\frac{\sum_u (r_{u,i}-\bar{r}_u)*(r_{u,j}-\bar{r}_u)}{\sqrt{\sum_u (r_{u,i}-\bar{r}_u)^2}*\sqrt{\sum_u (r_{u,j}-\bar{r}_u)^2}} \quad (2.1)$$

За U берем множество пользователей, которые оценили фильмы i и j, а $\acute{r}_u$ – средняя оценка, которую ставит пользователь u.

Также воспользуемся косинусной мерой схожести, которая рассчитывается по формуле:

$$¿(i,j)=\frac{\sum_u r_{u,i}*r_{u,j}}{\sqrt{\sum_u r_{u,i}}\sqrt{\sum_u r_{u,j}}} \quad (2.2)$$

Рейтинг для еще неоцененных фильмов в методе item-based посчитаем по этой формуле:

$$\widehat{r_{u,i}}=\frac{\sum_j ¿(i,j)*r_{u,j}}{\sum_j ¿(i,j)} \quad (2.3)$$

при $\widehat{r_{u,j}}\neq 0$

В методе user-based ищем похожих пользователей, используя меры схожести те же, что в предыдущем методе. Формула корреляции Пирсона:

$$¿(u,a)=\frac{\sum_i (r_{u,i}-\bar{r}_u)*(r_{a,i}-\bar{r}_a)}{\sqrt{\sum_i (r_{u,i}-\bar{r}_u)^2}*\sqrt{\sum_i (r_{a,i}-\bar{r}_a)^2}} \quad (2.4)$$

Косинусная мера сходства:

$$¿(u,a)=\frac{\sum_i r_{a,i}*r_{u,i}}{\sqrt{\sum_u r_{a,i}}\sqrt{\sum_u r_{u,i}}} \quad (2.5)$$

В этом методе для прогнозирования рейтинга воспользуемся формулой:

$$\widehat{r_{a,i}} = \frac{\sum_u r_{u,i} \times \sim(a,u)}{\sum_u ¿(a,u)} \quad (2.6)$$

Пересчет с учетом временного фактора осуществляется с помощью формулы:

$$r_u = r_u * e^{-d*(\Delta t)} \quad (2.7)$$

За Δt возьмем разницу между временем выставлением оценки и временем расчета прогноза.

Погрешность рассчитывалась по этим формулам:

$$\begin{matrix} r_{u,i} \vee ¿ \\ \Delta_i = ¿ r_{u,i} - \hat{¿} \end{matrix} \quad (2.8)$$

$$\varepsilon = \frac{1}{k} \sum_{i=1}^{k} \Delta_i \quad (2.9)$$

2.2 Диаграмма базы данных MovieLens

Таблицы данных были скачаны из интернета. Далее были загружены в MS SQL Server Management Studio. Из них составлена диаграмма, представленная на рисунке ниже. (Рисунок 2.1)

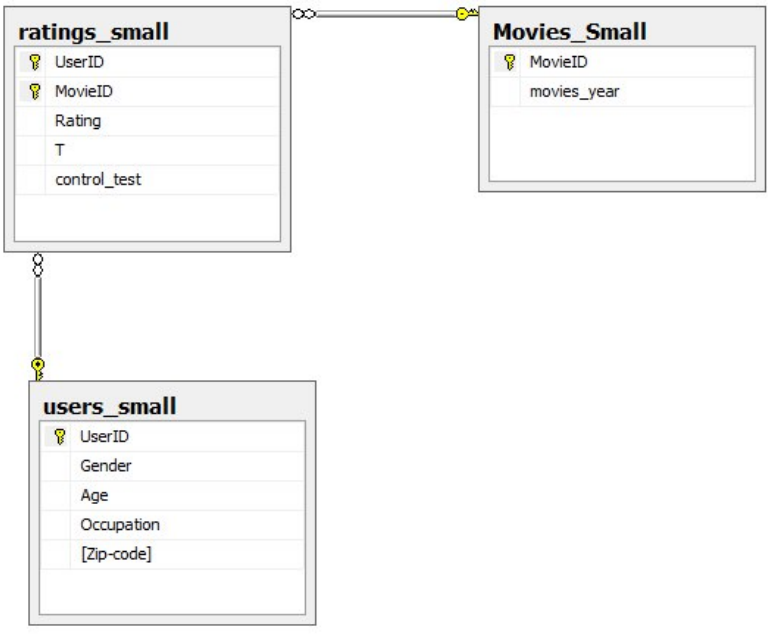

Рисунок 2.1 – диаграмма базы данных

Она состоит из трех таблиц: rating_small, users_small, Movies_Small.

В построенной диаграмме нет многозначных атрибутов и связей «многие ко многим». Выделены следующие отношения, представленные на таблицах ниже (таблицы 2.1, 2.2 и 2.3).

Таблица 2.1

rating_small (Рейтинги)

Имя Атрибута	Тип	Обязат
UserID	int	not null
MovieID	int	not null
Rating	float	not null
T	timestamp	not null
control_test	bit	not null

Таблица 2.2

users_small (Пользователи)

Имя Атрибута	Тип	Обязат
UserID	int	not null
Gender	nvarchar (5)	not null
Age	int	not null
Occupation	nvarchar(20)	not null
zip-code	int	not null

Таблица 2.3

Movies_Small (Фильмы)

Имя Атрибута	Тип	Обязат
MovieID	int	not null
movie_year	int	not null

2.3 Базовый алгоритм фильтрации

Пусть в нашей системе есть объекты и пользователи. Эти пользователи ставят оценки некоторым объектам. Нужно предположить, какую оценку пользователь поставит какому-либо объекту i.

Рассмотрим алгоритм item-based.

1. Используя формулы (2.1) и (2.2), для каждого объекта j находим, насколько он похож на объект i, для которого нужно спрогнозировать оценку.

2. Сортируем полученные результаты первого шага по убыванию и выбираем топ-К объектов, наиболее похожих на объект i.

3. Используя формулу (2.3) вычисляем прогнозируемую оценку для объекта i.

2.4 Алгоритм фильтрации с учетом временного фактора

Воспользуемся алгоритмами из предыдущего пункта, пересчитаем все шаги с формулой (2.7). За коэффициент d было решено взять число 0.00001, так как время T имеет тип timestamp, то есть секунды от 1 января 1970 года.

2.5 Выводы по разделу

В данном разделе была выполнена математическая постановка задачи, представлены все используемые формулы для расчета прогноза рейтингов, в том числе и формула, с помощью которой вводится временной фактор. Описан алгоритм, на основе которого проводились исследования для данной работы.

Также в этом разделе показана диаграмма базы данных, и описание всех полей в ее таблицах.

3 ИССЛЕДОВАНИЕ АЛГОРИТМА

3.1 Методика экспериментального исследования алгоритма

Весь эксперимент можно разделить на две части:

1. Исследование различия результатов в зависимости от изменения формулы меры близости объектов.

2. Исследование изменения результата в зависимости от того, учитываем мы или нет временной фактор в формулах.

Для этих исследований был выбран метод коллаборативной фильтрации item-based (основанный на сходстве объектов).

Вся выборка делится на тестовую и контрольную в процентном соотношение 80/20. В этом методе на первом шаге алгоритма используется формула для измерения сходства объектов. Для эксперимента было решено посчитать это сходство по двум разным формулам:

– формула косинусной меры;

– формула коэффициента корреляции Пирсона.

Затем сравнить результаты для того, чтобы посмотреть, как на них влияет изменение подхода в измерении коэффициента похожести и влияет ли вообще.

После завершения первой части эксперимента нужно этот метод пересчитать по тем же алгоритмам, но при этом учитывая временной фактор, то есть подставить новое (пересчитанное по новой формуле) значение рейтинга в основные формулы. Затем сравнить результаты и сделать выводы о том, как учтенный временной фактор влияет на ответ.

3.2 Программа для экспериментальных исследований

Все основные вычисления были выполнены через запросы в MS SQL Server Management Studio. Графики построены в Matlab.

Для вычисления меры сходства объектов были созданы дополнительные таблицы.

Запрос для заполнения таблицы с расчетом косинусной меры без учета временного фактора:

```
insert into Sim_CosItem(i_movieID, j_MovieID, sim)
select a.MovieID,b.MovieID,SUM(a.Rating*b.Rating)
from (select MovieID, UserID, Rating from dbo.ratings_small where control_test=1)a,
(select MovieID, UserID, Rating from dbo.ratings_small where control_test=1) b
where a.UserID=b.UserID
group by a.MovieID,b.MovieID
go
update dbo.Sim_CosItem set sim=sim/sqrt(b.S1)
from dbo.Sim_CosItem , (select MovieID, SUM(rating) as S1
```

```
                    from dbo.Ratings_small
                    where control_test=1
                    group by MovieID) b
where dbo.Sim_CosItem.i_MovieID=b.MovieID
go
update dbo.Sim_CosItem set sim=sim/sqrt(b.S1)
from dbo.Sim_CosItem , (select MovieID, SUM(rating) as S1
                    from dbo.Ratings_small
                    where control_test=1
                    group by MovieID) b
where dbo.Sim_CosItem.j_MovieID=b.MovieID
```

Запрос для заполнения таблицы с найденным коэффициентом корреляции Пирсона, не учитывая временной фактор:

```
insert into Sim_PirsItem(i_movieIDp, j_movieIDp, simPirsI, numerator)
select  a.MovieID,b.MovieID,SUM((a.Rating-c.avs)*(b.Rating-c.avs)),  SUM((a.Rating-
c.avs)*(b.Rating-c.avs))
  from (select MovieID, UserID, Rating from dbo.ratings_small where control_test=1)a ,
   (select MovieID, UserID, Rating from dbo.ratings_small where control_test=1)b ,
   (select avg(rating) as avs, UserID from dbo.ratings_Small where control_test=1 group
by UserID) c
  where a.UserID=b.UserID and c.UserID=a.UserID
group by a.MovieID,b.MovieID
go
update dbo.Sim_PirsItem set numerator=0,simPirsI=0
where simPirsI<0
go
update dbo.Sim_PirsItem set simPirsI=simPirsI/sqrt(c.numerator*c.numerator)
from (select MovieID, UserID, Rating
                from dbo.ratings_small
                where control_test=1) a, dbo.Sim_PirsItem c
where c.i_movieIDp=a.MovieID and  c.numerator<>0
go
update dbo.Sim_PirsItem set simPirsI=simPirsI/sqrt(c.numerator*c.numerator)
from (select MovieID, UserID, Rating
                from dbo.ratings_small
                where control_test=1) a, dbo.Sim_PirsItem c
where c.j_movieIDp=a.MovieID and c.numerator<>0
```

Запрос для заполнения таблицы с косинусной мерой с учетом временного фактора:

```
insert into dbo.Sim_CosTime(i_movieID, j_MovieID, sim)
select a.MovieID,b.MovieID,SUM(a.Tdelta*b.Tdelta)
from (select MovieID, UserID, Tdelta, control_test from dbo.RatingNew where
control_test=1) a,
```

```
(select MovieID, UserID, Tdelta from dbo.RatingNew where control_test=1) b
where a.UserID=b.UserID
group by a.MovieID,b.MovieID
go
update dbo.Sim_CosTime set sim=sim/sqrt(b.S1)
from dbo.Sim_CosTime , (select MovieID, SUM(Tdelta) as S1
                        from dbo.RatingNew
                        where control_test=1
                        group by MovieID) b
where dbo.Sim_CosTime.i_MovieID=b.MovieID
go
update dbo.Sim_CosTime set sim=sim/sqrt(b.S1)
from dbo.Sim_CosTime , (select MovieID, SUM(Tdelta) as S1
                        from dbo.RatingNew
                        where control_test=1
                        group by MovieID) b
where dbo.Sim_CosTime.j_MovieID=b.MovieID
```

Запрос для заполнения таблицы с найденным коэффициентом корреляции Пирсона, учитывая временной фактор:

```
insert into Sim_PirsTime(i_movieIDp, j_movieIDp, simPirsI, numerator)
select  a.MovieID,b.MovieID,SUM((a.Tdelta-c.avs)*(b.Tdelta-c.avs)),  SUM((a.Tdelta-
c.avs)*(b.Tdelta-c.avs))
  from (select MovieID, UserID, Tdelta from dbo.RatingNew2 where control_test=1) a,
   (select MovieID, UserID, Tdelta from dbo.RatingNew2 where control_test=1) b,
   (select avg(Tdelta) as avs, UserID from dbo.RatingNew2 where control_test=1 group
by UserID) c
  where a.UserID=b.UserID and c.UserID=a.UserID
group by a.MovieID,b.MovieID
go
update dbo.Sim_PirsTime set numerator=0,simPirsI=0
where simPirsI<0
go
update dbo.Sim_PirsTime set simPirsI=simPirsI/sqrt(c.numerator*c.numerator)
from (select MovieID, UserID, Tdelta
                from dbo.RatingNew2
                where control_test=1) a, dbo.Sim_PirsTime c
where c.i_movieIDp=a.MovieID and  c.numerator<>0
go
update dbo.Sim_PirsTime set simPirsI=simPirsI/sqrt(c.numerator*c.numerator)
from (select MovieID, UserID, Tdelta
                from dbo.RatingNew2
                where control_test=1) a, dbo.Sim_PirsTime c
where c.j_movieIDp=a.MovieID and c.numerator<>0
```

3.3 Результаты исследований

Эксперимент проводился на данных, разделенных на тестовую выборку и контрольную. На контрольной были подсчитаны погрешности работы алгоритма для разного количества фильмов. Числовые результаты первой части эксперимента представлены в таблице ниже (таблица 3.1).

Таблица 3.1

Сравнительная таблица погрешностей при разных мерах сходства двух объектов

k	Погрешность при косинусной мере сходства	k	Погрешность при корреляции Пирсона
60	0.6857718	60	0.5053989
80	0.5818117	80	0.4666446
100	0.5537641	100	0.4628906
120	0.4810318	120	0.3992219
140	0.4484705	140	0.3942962
160	0.4200267	160	0.3209039

Переходя ко второй части эксперимента, рассматривался вариант с использованием коэффициента корреляции Пирсона как меры сходства двух объектов, поскольку он оказался наиболее точным в данном случае. Результаты учета временного фактора представлены в таблице ниже (таблица 3.2).

Таблица 3.2

Сравнительная таблица погрешностей с учетом и без учета временного фактора

k	Погрешность без учета временного фактора	k	Погрешность с учетом временного фактора
60	0.5053989	60	0.3496937
80	0.4666446	80	0.3368896
100	0.4628906	100	0.2447144
120	0.3992219	120	0.2394329
140	0.3942962	140	0.1589404
160	0.3209039	160	0.1317049

Для обработки результатов исследований были использованы формулы для расчета погрешности (2.9). Чтобы построить графики, были посчитаны погрешности при разных объемах выборки пользователей. Для наглядности результаты представлены в виде графиков. Из этих графиков видно, что чем большее число схожих объектов берется для того, чтобы спрогнозировать оценку, тем более точные результаты получаются. (Рисунки 3.1 и 3.2)

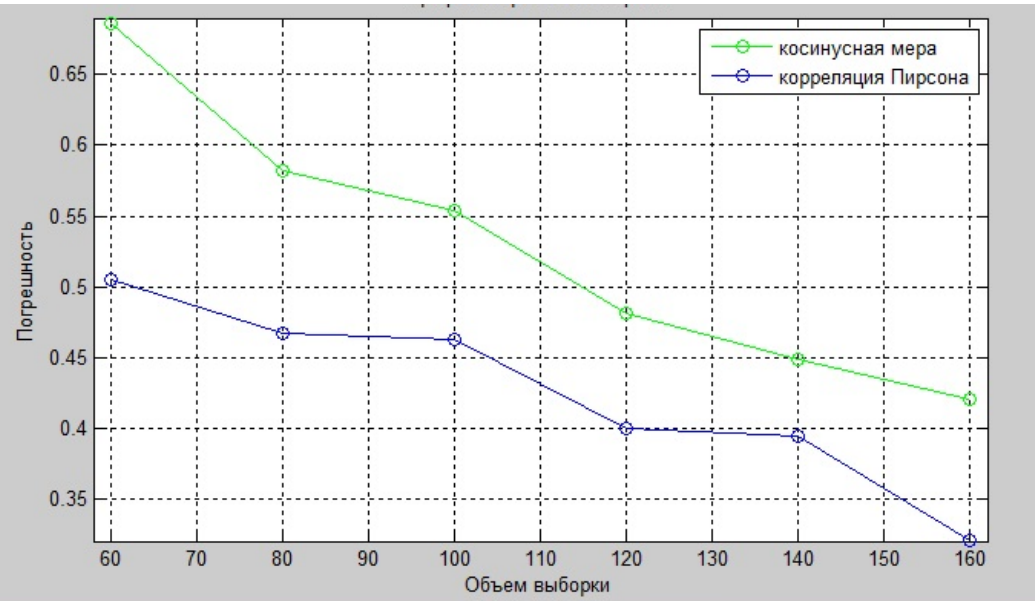

Рисунок 3.1 – График сравнения погрешности результатов

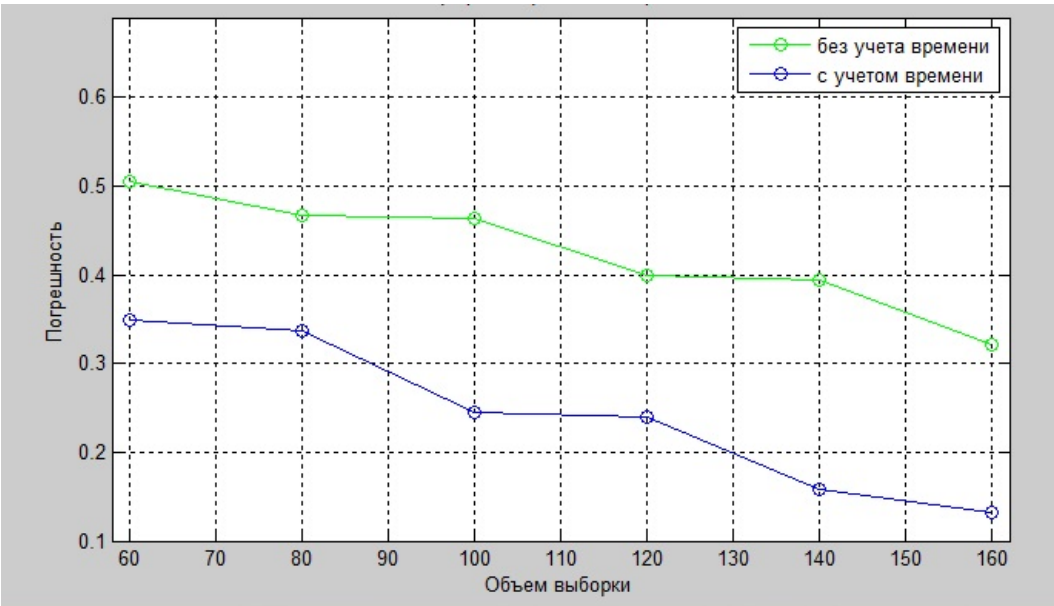

Рисунок 3.2 – График сравнения погрешностей результатов с учетом временного фактора и без учета

3.4 Рекомендации по улучшению алгоритма

Для улучшения данного алгоритма можно воспользоваться дополнительной информацией по пользователям, такой, как возраст, пол, специальность или, например, географическое положение (страна проживания). То есть рекомендуется объединение анамнестических и модельных методик.

3.5 Выводы по разделу

В данном разделе была сформулирована методика экспериментального исследования алгоритма, которая включает в себя две части: исследование влияния используемой формулы меры сходства на результаты, а также исследование влияния временного фактора на точность прогнозирования.

Разработана программа для экспериментальных исследований, проведено описание всех используемых скриптов.

ЗАКЛЮЧЕНИЕ

В ходе написания данной выпускной квалификационной работы был сделан обзор необходимой литературы и изучена предметная область. Была построена математическая модель алгоритма коллаборативной фильтрации, учитывающей временной фактор. Были исследованы две разные меры схожести двух объектов и для них построены графики погрешности прогнозирования. Также проведен сравнительный анализ для алгоритма с учетом времени и без его учета. В итоге получили, что временной фактор улучшает результаты прогнозирования.

По итогам выполнения выпускной квалификационной работы получена программа на языке T-SQL, которая вычисляет прогнозируемый рейтинг для фильмов из базы данных и считает погрешность полученной оценки, используя тестовую и контрольную выборки. Также получена программа в Matlab, которая наглядным образом, при помощи графиков, показывает сравнение полученных результатов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Гомзин, А.Г. Обзор рекомендательных систем и возможностей учета контекста при формировании индивидуальных рекомендаций / А.Г Гомзин // Academy – 2016 – 6(9) – С.20-22
2. Коллаборативная фильтрация [Электронный ресурс] URL: http://intellect.ml/kollaborativnaya-filtratsiya-4778 (Дата обращения 13.03.2017)
3. Мельник, К.В. Применение аппарата Байесовых сетей при обработке данных из медицинских карточек / К.В. Мельник, В.Н. Глушко // Science and Education a New Dimension: Natural and Technical Sciences. – I(2), Issue:15, 2013.– С.126-129
4. Пономарев, А. В. Обзор методов учета контекста в системах коллаборативной фильтрации / А.В. Пономарев //Труды СПИИРАН. – 2013. – Т. 7. – №. 30. – С. 169-188.
5. Рекомендательные системы [Электронный ресурс] URL: https://ru.wikipedia.org/wiki (Дата обращения 13.03.2017)
6. Федоровский А.Н. Архитектура рекомендательной системы, работающей на основе неявных пользовательских оценок / А.Н.Федоровский, В.К.Логачева//Труды 13й Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» - RCDL’2011 – 2011. – P.75-87
7. Abernethy J., A new approach to collaborative filtering: Operator estimation with spectral regularization / F. Bach., T. Evgeniou.// Journal of Machine Learning Research.- 2008.- Vol. 10.- P. 803-826
8. Adomavicius, G. Expert-Driven Validation of Rule-Based User Models in Personalization Applications/ G. Adomavicius, A. Tuzhilin // Data Mining and Knowledge Discovery, vol. 5, nos. 1 and 2, 2001. - P. 33-58,
9. Adomavicius, G. Multidimensional RecommenderSystems: A Data Warehousing Approach/ G. Adomavicius, A. Tuzhilin // Proc. Second Int’lWorkshop Electronic Commerce (WELCOM ’01), 2001b.
10. Ansari, A. Internet Recommendations Systems/A. Ansari // J. Marketing Research, Aug. 2000. - P. 363-375.
11. Balabanovíc, M. Fab: Content-Based, Collaborative Recommendation/ M. Balabanovíc, Y. Shoham // Communications of the ACM 40, - 1997. - P.66-72.
12. Brand, M. Fast online svd revisions for lightweight recommender systems/ M.Brand // In SIAM International Conference on Data Mining. - 2003. - P. 37-46
13. Billsus, D. Learning collaborative information filters/ D. Billsus, M. Pazzani // In Proceedings of the Fifteenth National Conference on Artificial Intelligence, CA. - 1998. - P.46-53.
14. Heckerman, D. Dependency Networks for Inference,Collaborative Filtering, and Data Visualization / D. Heckerman, D.M. Chickering, C. Meek, R. Rounthwaite, C.Kadie // Journal of Machine Learning Research 1, - 2000. - P.49-75.

15. Konstan, J. A. GroupLens: Applying Collaborative Filtering to Usenet news. /J.A. Konstan, B.N. Miller, D. Maltz, J. L. Herlocker, L.R. Gordon, J. Riedl//Communications of the ACM 40,- 1997. - P.77-87.

16. Linton, F. OWL: A Recommender System for Organziation-Wide Learning/ F.Linton, A. Charron // Proceedings of the 1998 Workshop on Recommender Systems, - 1998. - P.65-69.

17. Microsoft SQL Server 2008 [Электронный ресурс] URL: https://www.microsoft.com (Дата обращения 10.03.2017)

18. MovieLens 1M Dataset [Электронный ресурс] URL: https://grouplens.org/datasets/movielens/1m/ (Дата обращения 26.01.2017)

19. Melville, P. Recommender Systems/ P. Melville, V.Sindhwani // Encyclopedia of Machine Learning, Claude Sammut and Geoffrey Webb (Eds), Springer, 2010.

20. Robertson, S. Threshold Setting in Adaptive Filtering/ S.Robertson, S. Walker // J. Documentation, - vol. 56, - 2000. - P. 312-331.

21. Sarwar, B. Item-Based Collaborative Filtering Recommendation Algorithms/ B.Sarwar, G. Karypis, J. Konstan, J. Riedl // Proc. 10thInt'l WWW Conf., 2001.

22. Si, L. Flexible Mixture Model for CollaborativeFiltering / L. Si, R.Jin// Proc. 20th Int'l Conf. Machine Learning, Aug. 2003.

23. Soboroff, I. Combining Content and Collaborationin Text Filtering / I. Soboroff, C. Nicholas // Proc. Int'l Joint Conf. Artificial Intelligence Workshop: Machine Learning for Information Filtering, Aug. 1999.

24. Takacs, G. Scalable collaborative filtering approaches for large recommender systems/ G. Takacs, I. Pilaszy, B. Nemeth // Journal of Machine Learning Research (Special Topic on Mining and Learning with Graphs and Relations).- 2009.- Vol. 10.- P.623-656

ПРИЛОЖЕНИЕ 1

ТЕКСТ ПРОГРАММЫ

```
insert into Sim_CosItem(i_movieID, j_MovieID, sim)
select a.MovieID,b.MovieID,SUM(a.Rating*b.Rating)
from (select MovieID, UserID, Rating from dbo.ratings_small where control_test=1)a,
(select MovieID, UserID, Rating from dbo.ratings_small where control_test=1) b
where a.UserID=b.UserID
group by a.MovieID,b.MovieID
go
update dbo.Sim_CosItem set sim=sim/sqrt(b.S1)
from dbo.Sim_CosItem , (select MovieID, SUM(rating) as S1
                        from dbo.Ratings_small
                        where control_test=1
                        group by MovieID) b
where dbo.Sim_CosItem.i_MovieID=b.MovieID
go
update dbo.Sim_CosItem set sim=sim/sqrt(b.S1)
from dbo.Sim_CosItem , (select MovieID, SUM(rating) as S1
                        from dbo.Ratings_small
                        where control_test=1
                        group by MovieID) b
where dbo.Sim_CosItem.j_MovieID=b.MovieID

insert into Sim_PirsItem(i_movieIDp, j_movieIDp, simPirsI, numerator)
select  a.MovieID,b.MovieID,SUM((a.Rating-c.avs)*(b.Rating-c.avs)),  SUM((a.Rating-
c.avs)*(b.Rating-c.avs))
  from (select MovieID, UserID, Rating from dbo.ratings_small where control_test=1)a ,
   (select MovieID, UserID, Rating from dbo.ratings_small where control_test=1)b ,
   (select avg(rating) as avs, UserID from dbo.ratings_Small where control_test=1 group
by UserID) c
  where a.UserID=b.UserID and c.UserID=a.UserID
group by a.MovieID,b.MovieID
go
update dbo.Sim_PirsItem set numerator=0,simPirsI=0
where simPirsI<0
go
update dbo.Sim_PirsItem set simPirsI=simPirsI/sqrt(c.numerator*c.numerator)
from (select MovieID, UserID, Rating
                from dbo.ratings_small
                where control_test=1) a, dbo.Sim_PirsItem c
```

```
where c.i_movieIDp=a.MovieID and  c.numerator<>0
go
update dbo.Sim_PirsItem set simPirsI=simPirsI/sqrt(c.numerator*c.numerator)
from (select MovieID, UserID, Rating
                from dbo.ratings_small
                where control_test=1) a, dbo.Sim_PirsItem c
where c.j_movieIDp=a.MovieID and c.numerator<>0

insert into dbo.Sim_CosTime(i_movieID, j_MovieID, sim)
select a.MovieID,b.MovieID,SUM(a.Tdelta*b.Tdelta)
from (select MovieID, UserID, Tdelta, control_test from dbo.RatingNew where
control_test=1) a,
(select MovieID, UserID, Tdelta from dbo.RatingNew where control_test=1) b
where a.UserID=b.UserID
group by a.MovieID,b.MovieID
go
update dbo.Sim_CosTime set sim=sim/sqrt(b.S1)
from dbo.Sim_CosTime , (select MovieID, SUM(Tdelta) as S1
                        from dbo.RatingNew
                        where control_test=1
                        group by MovieID) b
where dbo.Sim_CosTime.i_MovieID=b.MovieID
go
update dbo.Sim_CosTime set sim=sim/sqrt(b.S1)
from dbo.Sim_CosTime , (select MovieID, SUM(Tdelta) as S1
                        from dbo.RatingNew
                        where control_test=1
                        group by MovieID) b
where dbo.Sim_CosTime.j_MovieID=b.MovieID

insert into Sim_PirsTime(i_movieIDp, j_movieIDp, simPirsI, numerator)
select  a.MovieID,b.MovieID,SUM((a.Tdelta-c.avs)*(b.Tdelta-c.avs)),  SUM((a.Tdelta-
c.avs)*(b.Tdelta-c.avs))
  from (select MovieID, UserID, Tdelta from dbo.RatingNew2 where control_test=1)a ,
   (select MovieID, UserID, Tdelta from dbo.RatingNew2 where control_test=1)b ,
    (select avg(Tdelta) as avs, UserID from dbo.RatingNew2 where control_test=1 group
by UserID) c
  where a.UserID=b.UserID and c.UserID=a.UserID
group by a.MovieID,b.MovieID
go
update dbo.Sim_PirsTime set numerator=0,simPirsI=0
where simPirsI<0
go
```

```
update dbo.Sim_PirsTime set simPirsI=simPirsI/sqrt(c.numerator*c.numerator)
from (select MovieID, UserID, Tdelta
            from dbo.RatingNew2
            where control_test=1) a, dbo.Sim_PirsTime c
where c.i_movieIDp=a.MovieID and  c.numerator<>0
go
update dbo.Sim_PirsTime set simPirsI=simPirsI/sqrt(c.numerator*c.numerator)
from (select MovieID, UserID, Tdelta
            from dbo.RatingNew2
            where control_test=1) a, dbo.Sim_PirsTime c
where c.j_movieIDp=a.MovieID and c.numerator<>0
```