# Сравнительное исследование совместной фильтрации Алгоритмы

Джунсок Ли, Минсюань Сан, Гай Ливан

14 мая 2012 г.

Абстрактный

Коллаборативная фильтрация — это быстро развивающаяся область исследований. Каждый год предлагается несколько новых методов, и пока не ясно, какие из них работают лучше всего и при каких условиях. В этой статье мы проводим исследование, сравнивая несколько методов коллаборативной фильтрации — как классических, так и современных — в различных экспериментальных контекстах. В частности, мы сообщаем выводы, контролирующие количество элементов, количество пользователей, уровень разреженности, критерии производительности и вычислительную сложность. Наши выводы определяют, какие алгоритмы работают хорошо и при каких условиях, и вносят вклад как в промышленное развертывание алгоритмов коллаборативной фильтрации, так и в исследовательское сообщество.

## 1 Введение

Совместная фильтрация — это быстро развивающаяся область исследований. Классические методы включают методы соседства, такие как основанная на памяти или основанная на пользователе совместная фильтрация. Более поздние методы часто вращаются вокруг факторизации матриц, включая разложение сингулярных значений и неотрицательную факторизацию матриц. Постоянно предлагаются новые методы, мотивированные экспериментами или теорией. Однако, несмотря на значительный исследовательский импульс, нет консенсуса или ясности относительно того, какой метод работает лучше всего.

Одной из трудностей, несомненно, являющейся центральной причиной отсутствия ясности, является то, что производительность различных методов существенно различается в зависимости от параметров проблемы. В частности, такие факторы, как количество пользователей, количество элементов и уровень разреженности (соотношение наблюдаемых оценок к общим) по-разному влияют на различные методы совместной фильтрации. Некоторые методы работают лучше в разреженных условиях, в то время как другие работают лучше в плотных условиях и т. д.

Существующие экспериментальные исследования в области коллаборативной фильтрации либо не сравнивают последние современные методы, либо не исследуют вариации относительно вышеупомянутых параметров проблемы. В этой статье мы делаем это и концентрируемся на сравнении как классических, так и последних современных методов. В наших экспериментах мы контролируем количество пользователей, количество элементов, уровень разреженности и рассматриваем множественную меру оценки и вычислительные затраты.

На основании проведенного нами сравнительного исследования мы приходим к следующим выводам.

1. Вообще говоря, методы, основанные на матричной факторизации, работают лучше всего с точки зрения точности предсказания. Тем не менее, в некоторых особых случаях, несколько других методов имеют явное преимущество перед методами матричной факторизации.

2. Точность прогнозирования различных алгоритмов зависит от количества пользователей, количества элементов и плотности, причем характер и степень зависимости различаются от алгоритма к алгоритму.

3. Существует сложная взаимосвязь между точностью прогноза, его дисперсией, временем вычислений и потреблением памяти, которая имеет решающее значение для выбора наиболее подходящего алгоритма рекомендательной системы.

В следующих разделах подробно описываются разработка и реализация экспериментального исследования, а также сами экспериментальные результаты.

# 2 Предыстория и сопутствующие работы

Прежде чем описывать наше экспериментальное исследование, мы кратко рассмотрим системы рекомендаций и методы совместной фильтрации.

## 2.1 Рекомендательные системы

В широком смысле, любая программная система, которая активно предлагает товар для покупки, подписки или инвестирования, может рассматриваться как рекомендательная система. В этом широком смысле, реклама также может рассматриваться как рекомендация. Однако мы в основном рассматриваем более узкое определение «персонализированной» рекомендательной системы, которая основывает рекомендации на информации, специфичной для пользователя.

Существует два основных подхода к системам персонализированных рекомендаций: фильтрация на основе контента и совместная фильтрация. Первый подход явно использует знания домена относительно пользователей или элементов. Знания домена могут соответствовать информации о пользователе, такой как возраст, пол, род занятий или местоположение [8], или информации об элементе, такой как жанр, производитель или продолжительность в случае рекомендации фильма.

Последняя категория совместной фильтрации (CF) не использует информацию о пользователе или элементе, за исключением частично наблюдаемой матрицы рейтинга. Матрица рейтинга содержит рейтинги элементов (столбцов) пользователями (строками) и обычно является двоичной, например, нравится/не нравится, или порядковой, например, от одной до пяти звезд в рекомендации фильма Netflix. Матрица рейтинга также может быть собрана неявно на основе активности пользователя, например, веб-поиск с последующим переходом по ссылке может быть интерпретирован как положительное оценочное суждение для выбранной гиперссылки [11]. В целом, матрица рейтинга крайне разрежена, поскольку маловероятно, что каждый пользователь испытал и предоставил оценки для всех элементов.

Гибридные стратегии совместной и контентной фильтрации объединяют два подхода, описанных выше, используя как матрицу рейтинга, так и информацию о пользователе и элементе. [26, 42, 27, 22, 15, 2, 27, 35]. Такие системы обычно достигают улучшенной точности прогнозирования по сравнению с системами контентной фильтрации и системами совместной фильтрации.

В этой статье мы сосредоточимся на сравнительном исследовании алгоритмов совместной фильтрации. Существует несколько причин, по которым не включена фильтрация на основе контента. Во-первых, серьезное сравнение систем совместной фильтрации само по себе является сложной задачей. Во-вторых, экспериментальные результаты фильтрации на основе контента тесно связаны с доменом и вряд ли будут переноситься из одного домена в другой. Методы совместной фильтрации, с другой стороны, используют только матрицу рейтинга, которая по своей природе схожа в разных доменах.

## 2.2 Совместная фильтрация

Системы коллаборативной фильтрации обычно делятся на две подгруппы: методы, основанные на памяти, и методы, основанные на моделях.

Методы на основе памяти просто запоминают матрицу рейтинга и выдают рекомендации на основе взаимосвязи между запрошенным пользователем и элементом и остальной частью матрицы рейтинга. Методы на основе модели подгоняют параметризованную модель под заданную матрицу рейтинга, а затем выдают рекомендации на основе подобранной модели.

Наиболее популярными методами CF на основе памяти являются методы на основе соседства, которые предсказывают рейтинги, ссылаясь на пользователей, чьи рейтинги похожи на рейтинги запрашиваемого пользователя, или на элементы, похожие на запрашиваемый элемент. Это мотивируется предположением, что если два пользователя имеют похожие рейтинги по некоторым элементам, они будут иметь похожие рейтинги по остальным элементам. Или, в качестве альтернативы, если два элемента имеют похожие рейтинги от части пользователей, два элемента будут иметь похожие рейтинги от остальных пользователей.

В частности, методы CF на основе пользователей [5] идентифицируют пользователей, похожих на запрашиваемого пользователя, и оценивают желаемый рейтинг как средний рейтинг этих похожих пользователей. Аналогично, методы CF на основе элементов [31] идентифицируют элементы, похожие на запрашиваемый элемент, и оценивают желаемый рейтинг как средний рейтинг этих похожих элементов. Методы соседства значительно различаются по тому, как они вычисляют средневзвешенное значение рейтингов. Конкретные примеры мер сходства, которые влияют на усредняющие веса, включают корреляцию Пирсона, векторный косинус и среднеквадратичную разность (MSD). Методы, основанные на соседстве, можно расширить с помощью голосов по умолчанию, обратной частоты пользователя и усиления случая [5]. Недавний метод, основанный на соседстве [37], создает оценщик плотности ядра для неполных частичных ранжирований и предсказывает рейтинги, которые минимизируют апостериорные потери.

С другой стороны, методы на основе моделей подгоняют параметрическую модель под данные обучения, которые впоследствии можно использовать для прогнозирования невидимых рейтингов и выдачи рекомендаций. Методы на основе моделей включают кластерную CF [38, 6, 7, 32, 40], байесовские классификаторы [23, 24] и методы на основе регрессии [39]. Метод наклона один [20] подгоняет линейную модель под матрицу рейтинга, достигая быстрых вычислений и разумной точности.

Недавний класс успешных моделей CF основан на факторизации матриц низкого ранга. Регуляризованный метод SVD [4] факторизует матрицу рейтинга в произведение двух матриц низкого ранга (профиль пользователя и профиль элемента), которые используются для оценки отсутствующих записей.

Альтернативный метод — неотрицательная матричная факторизация (NMF) [18], которая отличается тем, что ограничивает матрицы низкого ранга, формирующие факторизацию, чтобы они имели неотрицательные записи. Недавние вариации — вероятностная матричная факторизация (PMF) [30], байесовский PMF [29], нелинейный PMF [17], матричная факторизация с максимальным запасом (MMMF) [34, 28, 9] и нелинейный главный компонентный анализ (NPCA) [41].

## 2.3 Меры оценки

Наиболее распространенной мерой оценки точности прогнозирования КФ являются средняя абсолютная ошибка (MAE) и корень средней квадратической ошибки (RMSE):

$$МАЭ = \frac{1}{N} \sum_{у,я} |п_{у,я} - г_{у,я}|$$

$$СКО = \sqrt{\frac{1}{N} \sum_{у,я} (п_{у,я} - г_{у,я})^2}$$

где $п_{у,я}$ и $г_{у,я}$ являются прогнозируемым и наблюдаемым рейтингом для пользователя *ты* и предмет *я*, соответственно. Сумма выше охватывает маркированный набор, который отложен для целей оценки (тестовый набор). Другие меры оценки — это точность, полнота и меры F1 [5].

Гунавардана и Шани [10] утверждают, что разные метрики оценки приводят к разным выводам относительно относительной производительности алгоритмов CF. Однако большинство исследовательских работ по CF мотивируют свой алгоритм, исследуя одну меру оценки. В этой статье мы рассматриваем производительность разных алгоритмов CF как функцию параметров задачи, измеренную с использованием нескольких разных критериев оценки.

### 2.4 Сопутствующая работа

Доступно несколько хорошо написанных обзоров по системам рекомендаций. Адомавичус и Тужилин [1] классифицировали алгоритмы CF, доступные по состоянию на 2006 год, на основанные на контенте, совместные и гибридные и обобщили возможные расширения. Су и Хошгофтаар [36] больше сосредоточились на методах CF, включая основанные на памяти, основанные на моделях и гибридные методы. Этот обзор содержит большинство современных алгоритмов, доступных по состоянию на 2009 год, включая конкурентов Netflix Prize. Недавний учебник по системам рекомендаций знакомит с традиционными методами и исследует дополнительные вопросы, такие как проблемы конфиденциальности [13].

Доступно несколько экспериментальных исследований. Первое исследование Бриза и др. [5] сравнило два популярных метода на основе памяти (корреляция Пирсона и векторное сходство) и два классических метода на основе моделей (кластеризация и байесовская сеть) на трех разных наборах данных. Более позднее экспериментальное сравнение алгоритмов CF [12] сравнивает CF на основе пользователя, CF на основе элемента, SVD и несколько других методов на основе моделей, фокусируясь на приложениях электронной коммерции. Оно рассматривает точность, отзыв, F1-меру и ранговую оценку в качестве мер оценки с комментариями о проблеме вычислительной сложности. Однако это игнорирует некоторые стандартные меры оценки, такие как MAE или RMSE.

## 2.5 Премия Netflix и набор данных

Конкуренция Netflix[1]проводился с октября 2006 года по июль 2009 года, когда команда BellKor Pragmatic Chaos выиграла приз в миллион долларов. Целью этого конкурса было повышение точности прогнозирования (в терминах RMSE) системы рекомендаций фильмов Netflix на 10%. Победившая команда использовала гибридный метод, который использовал временную динамику для учета дат, в которых сообщались рейтинги [16, 3]. Второе место, The Ensemble [33], достигло сопоставимых с победившей командой результатов, путем линейного объединения большого количества моделей.

Хотя конкурс Netflix завершился, набор данных, использованный в этом конкурсе, по-прежнему используется в качестве стандартного набора данных для оценки методов CF. Он содержит 480 046 пользователей и 17 770 элементов с 95 947 878 оценками. Это представляет собой разреженный уровень 1,12% (общее количество записей, деленное на наблюдаемые записи). Более старые и меньшие стандартные наборы данных включают MovieLens (6 040 пользователей, 3 500 элементов с 1 000 000 оценок), EachMovie (72 916 пользователей, 1 628 элементов с 2 811 983 оценками) и BookCrossing (278 858 пользователей, 271 379 элементов с 1 149 780 оценками).

# 3 Экспериментальное исследование

Ниже мы опишем некоторые подробности нашего экспериментального исследования, а затем перейдем к описанию наших основных результатов.

## 3.1 Экспериментальный дизайн

Для проведения наших экспериментов и облегчения их воспроизводимости мы внедрили PREA[2]набор инструментов, реализующий 15 алгоритмов, перечисленных в таблице 1. Набор инструментов доступен для публичного использования и будет обновляться дополнительными современными алгоритмами, предложенными исследовательским сообществом.

В Таблице 1 есть три элементарных базовых уровня: постоянная функция (идентичное предсказание для всех пользователей и всех элементов), среднее значение пользователя (постоянное предсказание для каждого пользователя на основе их средних оценок) и среднее значение элемента (постоянное предсказание для каждого элемента на основе их средних оценок). Методы на основе памяти, перечисленные в Таблице 1, являются классическими методами, которые хорошо работают и часто используются в коммерческих условиях. Методы, перечисленные в категориях факторизации матрицы и других, являются более современными методами, предложенными в исследовательской литературе.

В наших экспериментах мы использовали набор данных Netflix, стандартный бенчмарк в литературе по CF, который больше и новее альтернативных бенчмарков. Чтобы облегчить измерение зависимости между точностью прогнозирования и размером и плотностью набора данных, мы отсортировали матрицу рейтинга так, чтобы ее строки и столбцы были перечислены в порядке убывания уровня плотности. Затем мы реализовали определенный шаблон разреженности, выбрав верхний $k$ряды и $l$столбцов и подвыборки для достижения требуемой разреженности.

[1]http://www.netflixprize.com
[2]http://prea.gatech.edu

| Категория | Алгоритмы |
|---|---|
| Базовый уровень | Постоянный<br>Средний пользователь<br>Средний показатель по предмету |
| Память<br>- основанный на | На основе пользователя [36]<br>На основе пользователя с параметрами по умолчанию<br>[5] На основе элементов [31]<br>На основе элементов с Default |
| Матрица Факторизация<br>- основанный на | Регуляризированный SVD [25]<br>НМФ [19]<br>ПМФ [30]<br>Байесовский PMF [29]<br>Нелинейный PMF [17] |
| Другие | Склон-один [20]<br>НПКА [41]<br>Ранговый CF [37] |

Таблица 1: Список алгоритмов рекомендаций, использованных в экспериментах

User Count

| Item Count | 500 | 1000 | 1500 | 2000 | 2500 | 3000 | 3500 | 4000 | 4500 | 5000 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1000 | 82.0% | 74.3% | 68.3% | 62.8% | 58.2% | 54.0% | 50.5% | 47.2% | 44.3% | 41.8% |
| 2000 | 80.0% | 71.5% | 64.7% | 58.8% | 53.8% | 49.4% | 45.7% | 42.4% | 39.5% | 37.0% |
| 3000 | 78.5% | 69.4% | 62.3% | 56.1% | 51.0% | 46.6% | 42.9% | 39.6% | 36.7% | 34.2% |
| 4000 | 77.3% | 67.8% | 60.4% | 54.1% | 48.9% | 44.5% | 40.8% | 37.5% | 34.7% | 32.2% |
| 5000 | 76.4% | 66.5% | 58.9% | 52.6% | 47.3% | 42.8% | 39.2% | 35.9% | 33.2% | 30.8% |
| 6000 | 75.6% | 65.4% | 57.7% | 51.2% | 46.0% | 41.5% | 37.8% | 34.6% | 31.9% | 29.6% |
| 7000 | 74.8% | 64.4% | 56.5% | 50.1% | 44.8% | 40.3% | 36.7% | 33.5% | 30.9% | 28.6% |
| 8000 | 74.0% | 63.5% | 55.5% | 49.0% | 43.7% | 39.3% | 35.7% | 32.6% | 29.9% | 27.7% |
| 9000 | 73.3% | 62.6% | 54.5% | 48.0% | 42.7% | 38.4% | 34.8% | 31.7% | 29.1% | 26.9% |
| 10000 | 72.7% | 61.8% | 53.7% | 47.1% | 41.9% | 37.5% | 34.0% | 31.0% | 28.4% | 26.2% |

User Count

| Item Count | 500 | 1000 | 1500 | 2000 | 2500 | 3000 | 3500 | 4000 | 4500 | 5000 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1000 | 82.0% | 66.7% | 56.1% | 46.6% | 39.8% | 33.1% | 29.3% | 24.3% | 21.2% | 18.7% |
| 2000 | 78.0% | 59.2% | 46.2% | 35.6% | 28.1% | 21.7% | 18.0% | 13.8% | 11.3% | 9.4% |
| 3000 | 75.5% | 55.2% | 41.6% | 31.1% | 23.8% | 17.9% | 14.6% | 11.0% | 8.9% | 7.3% |
| 4000 | 73.5% | 52.1% | 38.5% | 28.1% | 21.0% | 15.5% | 12.4% | 9.2% | 7.5% | 6.1% |
| 5000 | 72.9% | 50.3% | 36.3% | 25.9% | 19.1% | 14.0% | 11.1% | 8.1% | 6.3% | 5.2% |
| 6000 | 71.4% | 48.4% | 34.0% | 24.0% | 17.6% | 12.8% | 9.9% | 7.2% | 5.8% | 4.7% |
| 7000 | 70.3% | 46.7% | 32.5% | 22.7% | 16.2% | 11.6% | 9.0% | 6.6% | 5.2% | 4.2% |
| 8000 | 68.6% | 44.8% | 30.9% | 21.2% | 15.1% | 10.9% | 8.4% | 6.1% | 4.9% | 3.9% |
| 9000 | 67.8% | 43.6% | 29.6% | 20.3% | 14.4% | 10.2% | 7.8% | 5.7% | 4.5% | 3.6% |
| 10000 | 66.8% | 42.2% | 28.3% | 19.2% | 13.6% | 9.8% | 7.4% | 5.3% | 4.2% | 3.3% |

Рисунок 1: Плотность рейтинга (кумулятивная на левой панели и некумулятивная на правой панели) в матрице рейтинга Netflix, отсортированная по убыванию плотности строк и столбцов. Подробнее см. в тексте.

На рисунке 1 показан уровень плотности отсортированной матрицы рейтинга. Например, верхний правый угол отсортированной матрицы рейтинга, содержащей 5000 лучших пользователей и 2000 лучших элементов, имеет 52,6% плотности. Другими словами, 47,4% оценок отсутствуют. Плотность всего набора данных составляет около 1%. Мы подвыбираем заданный уровень плотности, который будет использоваться для обучения, а также 20% для целей тестирования. Мы перекрестно проверяем каждый эксперимент 10 раз с различными разделениями обучения и тестирования. Эксперименты проводились на двухпроцессорном процессоре Intel Xeon X5650 (6 ядер, 12 потоков, 2,66 ГГц) с 96 ГБ основной памяти.

## 3.2 Зависимость от размера и плотности данных

Начнем с исследования зависимости точности прогнозирования от размера набора данных (количества пользователей и количества элементов) и от плотности рейтинга. Особый интерес представляет изменчивость этой зависимости в разных алгоритмах CF. Эта изменчивость является ключом к определению того, какие алгоритмы CF следует использовать в конкретной ситуации. Ниже мы начнем с рассмотрения одномерной зависимости точности прогнозирования от каждого из этих трех

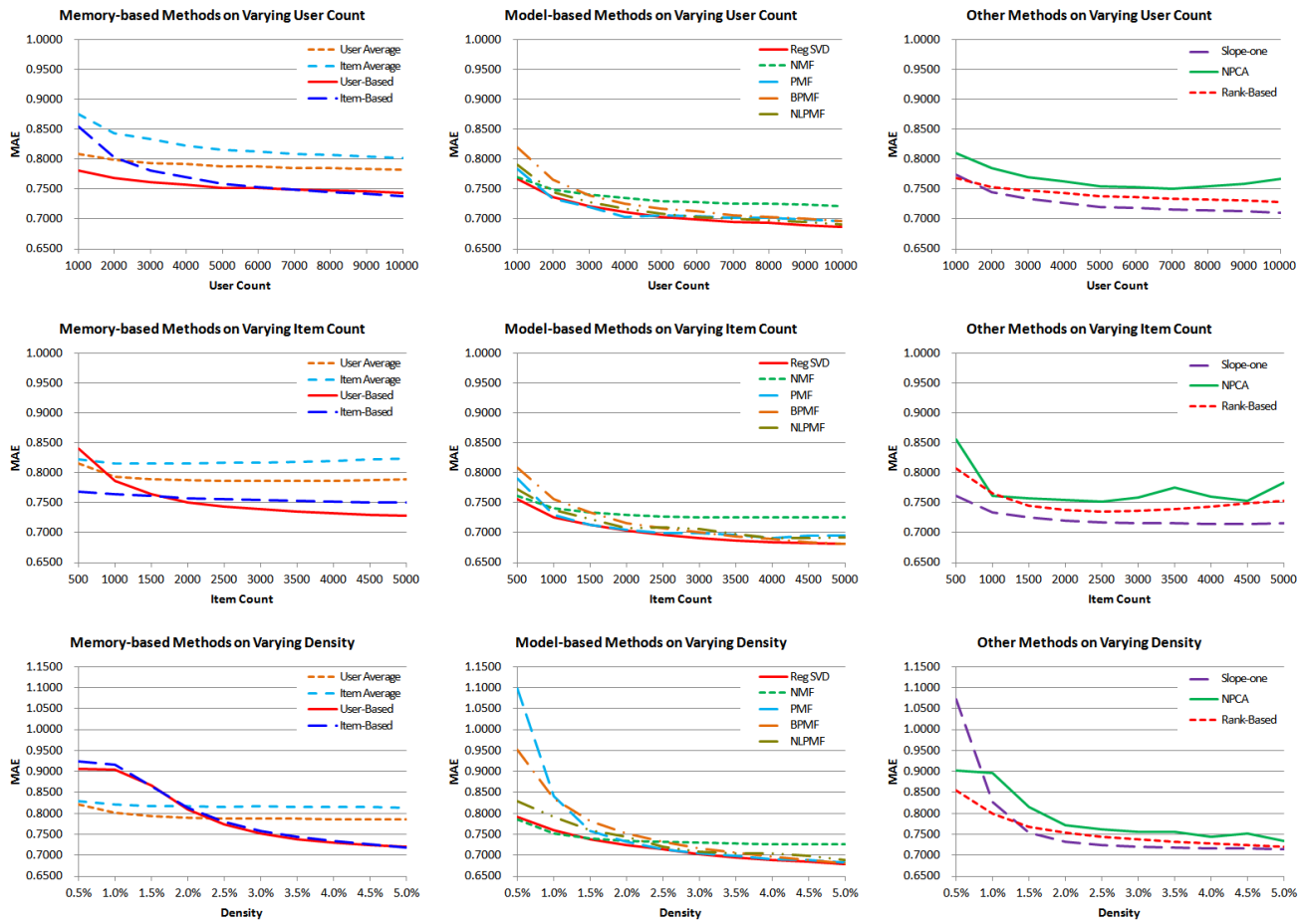


Рисунок 2: Прогнозируемая потеря как функция количества пользователей (вверху), количества элементов (в середине), плотности (внизу).

количества: количество пользователей, количество элементов и уровень плотности. Затем мы завершаем исследованием многомерной зависимости между точностью прогнозирования и этими тремя переменными.

### 3.2.1 Зависимость от количества пользователей

Рисунок 2 (верхний ряд) представляет собой график зависимости средней абсолютной ошибки (MAE) от количества пользователей, при этом каждая из трех панелей фокусируется на методах CF в определенной категории (основанные на памяти и базовые значения, основанные на модели и другие). Количество элементов и плотность были зафиксированы на уровне 2000 и 3% соответственно. Мера оценки RMSE показывает очень похожую тенденцию.

Мы исключили простейшую базовую линию правила постоянного прогнозирования, поскольку ее производительность намного хуже, чем у других. Варианты голосования по умолчанию методов CF на основе пользователя и на основе элемента не дали заметных изменений (мы изобразили на графике вариант, который работал лучше всего).

Чтобы сравнить, как различные алгоритмы зависят от количества пользователей, мы подогнали модель линейной регрессии к кривым на рисунке 2 (верхний ряд). Наклон $m$ и перехватывать $b$ Коэффициенты регрессии представлены в Таблице 2. Интерсепт $b$ указывает на ожидаемую потерю MAE алгоритма, когда число пользователей приближается к 0. Наклон $m$ указывает на скорость распада MAE.

Рассматривая рисунок 2 и таблицу 2, мы делаем следующие наблюдения.

1. Методы факторизации матриц в целом показывают лучшую производительность, когда число пользователей становится достаточно большим (>3,000).

2. В целом, наиболее эффективным алгоритмом является регуляризованный SVD.

3. Когда число пользователей достаточно мало, разница между методами факторизации матриц и более простыми методами, основанными на соседстве, очень мала.

4. Усредненные по элементам, основанные на элементах, регуляризованные SVD, PMF, BPMF и NLPMF, как правило, наиболее чувствительны к изменению количества пользователей.

5. Постоянный базовый уровень, среднее значение пользователя, основанный на пользователе, NMF, NPCA и основанный на ранге относительно нечувствительны к количеству пользователей.

6. Существует существенная разница в чувствительности между двумя популярными методами, основанными на соседстве: CF, основанным на пользователях, и CF, основанным на элементах.

7. CF на основе пользователя чрезвычайно эффективен для небольшого количества пользователей, но имеет почти постоянную зависимость от количества пользователей. CF на основе элемента сначала работает значительно хуже, но превосходит все другие методы на основе памяти для большего количества пользователей.

### 3.2.2 Зависимость от количества элементов

По аналогии с разделом 3.2.1 мы исследуем здесь зависимость потери предсказания от количества элементов, фиксируя количество пользователей на уровне 5000 и плотность на уровне 3% и 3% соответственно. Рисунок 2 (средний ряд) показывает MAE как функцию количества элементов для трех различных категорий алгоритмов CF. Таблица 2 показывает коэффициенты регрессии (см. описание в разделе 3.2.1).

Рассматривая рисунок 2 и таблицу 2, мы делаем следующие наблюдения, которые в значительной степени согласуются с наблюдениями в разделе 3.2.1.

1. Методы факторизации матриц в целом показывают лучшую производительность, когда количество элементов становится достаточно большим (>1,000).

2. В целом, наиболее эффективным алгоритмом является регуляризованный SVD.

3. Когда число пользователей достаточно мало, разница между методами факторизации матриц и более простыми методами, основанными на соседстве, очень мала.

| Алгоритм | Количество пользователей | | Количество предметов | | Плотность | |
|---|---|---|---|---|---|---|
| | $10^{-6}м$ | $б$ | $10^{-6}м$ | $б$ | $м$ | $б$ |
| Постоянный | - 1,7636 | 0,9187 | - 13.8097 | 0,9501 | - 0,0691 | 0,9085 |
| Средний пользователь | - 2,5782 | 0,8048 | - 3,5818 | 0,8006 | - 0,6010 | 0,8094 |
| Средний показатель по предмету | - 6.6879 | 0,8593 | + 1.1927 | 0,8155 | - 0,2337 | 0,8241 |
| На основе пользователя | - 3,9206 | 0,7798 | - 18.1648 | 0,8067 | - 4,7816 | 0,9269 |
| На основе пользователя (значения по умолчанию) На | - 3,6376 | 0,7760 | - 19.3139 | 0,8081 | - 4,7081 | 0,9228 |
| основе элемента | - 10.0739 | 0,8244 | - 3.2230 | 0,7656 | - 4.7104 | 0,9255 |
| На основе элементов (значения по умолчанию) | - 10,4424 | 0,8271 | - 3,6473 | 0,7670 | - 4,9147 | 0,9332 |
| Наклон один | - 5,6624 | 0,7586 | - 7,5467 | 0,7443 | - 5.1465 | 0,9112 |
| Регуляризированный СВД | - 7,6176 | 0,7526 | - 14.1455 | 0,7407 | - 2.2964 | 0,7814 |
| НМФ | - 4,4170 | 0,7594 | - 5,7830 | 0,7481 | - 0,9792 | 0,7652 |
| ПМФ | - 6.9000 | 0,7531 | - 14.7345 | 0,7529 | - 6.3705 | 0,9364 |
| Байесовский PMF | - 11.0558 | 0,7895 | - 23.7406 | 0,7824 | - 4,9316 | 0,8905 |
| Нелинейный PMF | - 8,8012 | 0,7664 | - 14.7588 | 0,7532 | - 2,8411 | 0,8135 |
| НПКА | - 4.1497 | 0,7898 | - 7.2994 | 0,7910 | - 3,5036 | 0,8850 |
| Ранговый CF | - 3,8024 | 0,7627 | - 7.3261 | 0,7715 | - 2,4686 | 0,8246 |

Таблица 2: Коэффициенты регрессии $y=мx+б$ для кривых на рисунке 2. Переменная $x$ представляет собой количество пользователей, количество элементов или плотность, а переменная $y$ представляет собой потерю прогнозирования MAE.

4. Регуляризированные SVD, PMF, BPMF и NLPMF, основанные на пользователях, как правило, наиболее чувствительны к изменению количества элементов.

5. Средний показатель по предметам, показатель на основе предметов и показатель NMF, как правило, менее чувствительны к изменению количества предметов.

6. Существует существенная разница в чувствительности между двумя популярными методами, основанными на соседстве: CF, основанным на пользователях, и CF, основанным на элементах.

7. CF на основе элементов чрезвычайно эффективен для небольшого количества элементов, но имеет почти постоянную зависимость от количества элементов. CF на основе пользователей сначала работает значительно хуже, но значительно превосходит все другие методы на основе памяти для большего количества пользователей.

8. В сочетании с наблюдениями в разделе 3.2.1 мы приходим к выводу, что slope-one, NMF и NPCA нечувствительны к изменениям как в количестве пользователей, так и в количестве элементов. PMF и BPMF относительно чувствительны к изменениям как в количестве пользователей, так и в количестве элементов.

### 3.2.3 Зависимость от плотности

На рисунке 2 (нижний ряд) показана зависимость потерь MAE от плотности рейтинга, при этом количество пользователей и элементов зафиксировано на уровне 5000 и 2000. Как и в разделе 3.2.1, в таблице 2 показаны коэффициенты регрессии, соответствующие производительности при плотности, близкой к 0, и чувствительность функции MAE к уровню плотности.

Рассматривая Рисунок 2 (нижний ряд) и Таблицу 2, мы делаем следующие наблюдения.

1. Простые базовые показатели (среднее значение пользователя и среднее значение элемента) прекрасно работают при низкой плотности.

2. Наиболее эффективными алгоритмами, по-видимому, являются регуляризованные SVD.

3. CF на основе пользователя и CF на основе элемента показывают замечательную схожую зависимость от уровня плотности. Это резко контрастирует с их различными зависимостями от пользователя и количества элементов.

4. По мере увеличения уровня плотности различия в точности прогнозирования различных алгоритмов сокращаются.

5. Методы, основанные на пользователе, на элементе, наклонные, PMF и BPMF в значительной степени зависят от плотности.

6. Три базовых уровня и NMF относительно независимы от плотности.

7. Производительность slope-one и PMF значительно ухудшается при низких плотностях, хуже самых слабых базовых линий. Тем не менее, оба алгоритма демонстрируют выдающуюся производительность при высоких плотностях.

### 3.2.4 Многомерные зависимости между потерей прогноза, количеством пользователей, количеством элементов и плотностью

Одномерные зависимости, рассмотренные ранее, показывают важные тенденции, но ограничены, поскольку они изучают изменчивость одной величины, фиксируя другие величины на произвольных значениях. Теперь мы переходим к изучению зависимости между потерей прогноза и следующими переменными: количество пользователей, количество элементов и плотность. Мы делаем это, строя график MAE как функцию количества пользователей, количества элементов и плотности (рисунок 3–4) и подгоняя многомерные регрессионные модели к зависимости MAE от количества пользователей, количества элементов и плотности (таблица 3).

На рисунке 3–4 показаны контуры равной высоты MAE в зависимости от количества пользователей ($x$ ось) и количество элементов ($y$ось) для нескольких уровней плотности (горизонтальные панели) и для нескольких алгоритмов CF (вертикальные панели). Обратите внимание, что все контурные графики имеют одинаковую$x$и$y$шкалы осей и так напрямую сопоставимы. Интервалы между различными контурными линиями представляют разницу в 0,01 в MAE и поэтому больше контурных линий представляют более высокую зависимость от$x$ и$y$Ось. Аналогичные графики RMSE показывают схожую тенденцию с этими графиками MAE.

В таблице 3 показаны коэффициенты регрессии.$м_{ты}, м_я, м_г$соответствующий линейной модели

$$y = м_{ты} x_{ты} + м_я x_я + м_г x_г + б \quad (1)$$

где$x_{ты}, x_я$, и$x_г$указать количество пользователей, количество элементов и плотность,$б$— постоянный член, и$y$это МАЭ.

На основании рисунков 3–4 и таблицы 3 мы делаем следующие наблюдения.

1. Одномерные связи, обнаруженные в предыдущем разделе для фиксированных значений оставшихся двух переменных, не обязательно выполняются в общем случае. Например, вывод о том, что PMF относительно чувствителен к количеству пользователей и количеству элементов при уровне разреженности 1% и уровне разреженности 3%, больше не действителен для уровней плотности 5%. Таким образом, важно провести многомерный, а не одномерный анализ зависимости потери прогнозирования от параметров задачи.

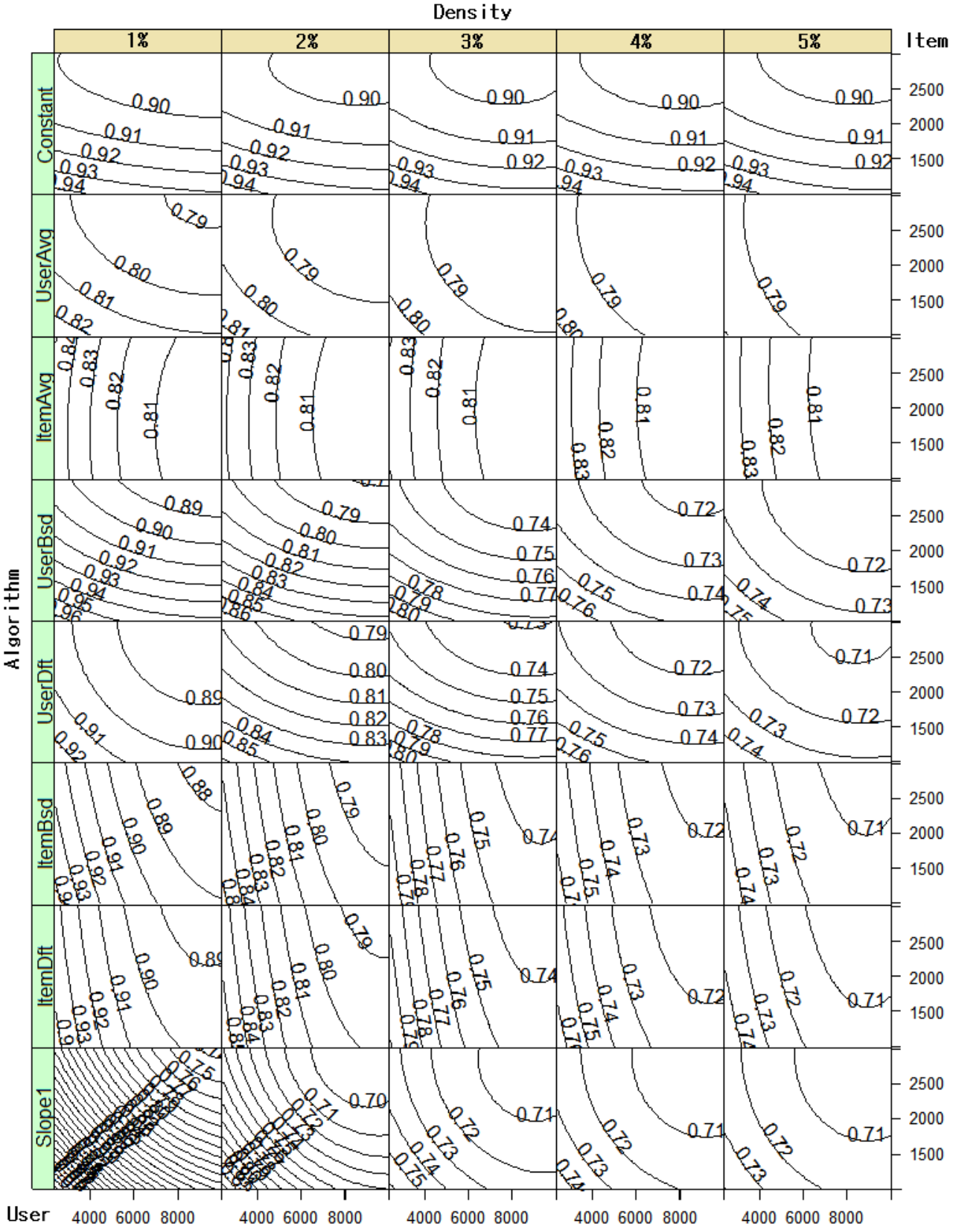


Рисунок 3: Контуры MAE для простого метода1d1 (более низкие значения означают лучшую производительность.)

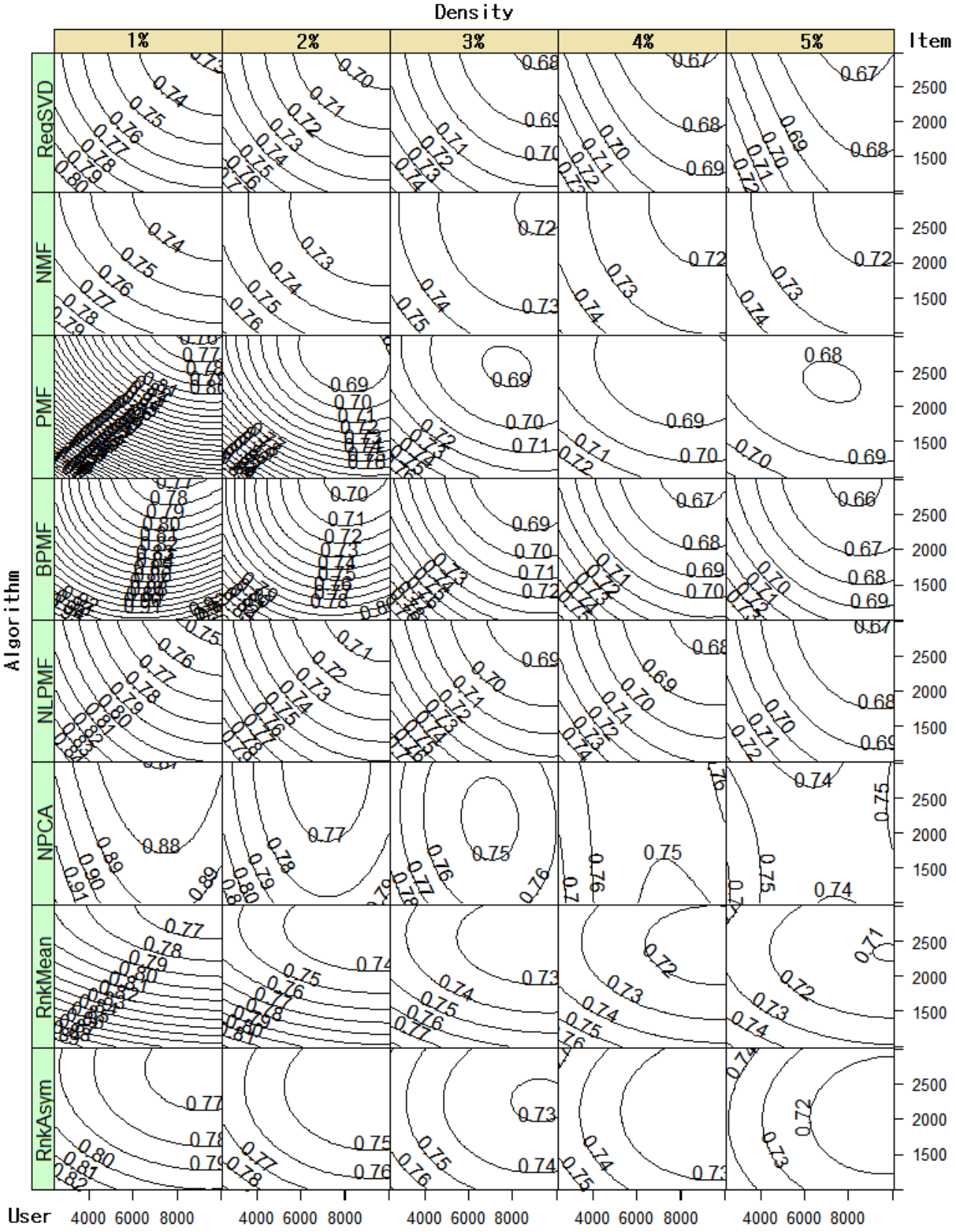


Рисунок 4: Контуры MAE для усовершенствованного метамфетамина12od (более низкие значения означают лучшую производительность.)

2. Форма контурных кривых MAE различается от алгоритма к алгоритму. Например, контурные кривые константы, среднего пользователя, основанного на пользователе и основанного на пользователе со значениями по умолчанию горизонтальны, что означает, что эти алгоритмы в значительной степени зависят от количества элементов, независимо от количества пользователей. С другой стороны, средний элемент, основанный на элементе и основанный на элементе со значениями по умолчанию показывают вертикальные контурные линии, что показывает высокую зависимость от количества пользователей.

3. Более высокая зависимость от количества пользователей и количества элементов коррелирует с высокой зависимостью от плотности.

4. В последнем столбце Таблицы 3 суммированы тенденции зависимости, основанные на абсолютном значении коэффициентов регрессии и их ранге. Вообще говоря, базовые линии относительно нечувствительны, методы на основе памяти зависят от одной переменной (вопреки их названиям), а методы факторизации матриц сильно зависят как от размера набора данных, так и от плотности.

| Алгоритм | $м_{ты}$ | $м_{я}$ | $м_{г}$ | $б$ | Краткое содержание |
|---|---|---|---|---|---|
| Постоянный | (15) -0,0115 | (8) -0,0577 | (15) -0,0002 | 0.9600 | Еженедельно зависит от всех переменных. |
| Средний пользователь | (13) -0,0185 | (14) -0,0164 | (13) -0,0188 | 0,8265 | Еженедельно зависит от всех переменных. |
| Средний показатель по предмету | (8) -0,0488 | (15) +0,0003 | (14) -0,0078 | 0,8530 | Еженедельно зависит от всех переменных. |
| На основе пользователя | (11) -0,0282 | (3) -0,0704 | (1) -0,2310 | 0,9953 | Еженедельно зависит от количества пользователей. |
| На основе пользователя (по умолчанию) | (12) -0,0260 | (7) -0,0598 | (5) -0,2185 | 0,9745 | Еженедельно зависит от количества пользователей. |
| На основе элементов | (3) -0,0630 | (12) -0,0172 | (4) -0,2201 | 0,9688 | Еженедельно зависит от количества элементов. |
| На основе элементов (с параметрами по умолчанию) | (2) -0,0632 | (13) -0,0167 | (2) -0,2286 | 0,9751 | Еженедельно зависит от количества элементов. |
| Склон-один | (1) -0,0746 | (4) -0,0702 | (8) -0,1421 | 0,9291 | Сильно зависит от всех переменных. Сильно зависит |
| Регуляризированный СВД | (7) -0,0513 | (9) -0,0507 | (11) -0,0910 | 0,8371 | от размера набора данных. Еженедельно зависит от |
| НМФ | (9) -0,0317 | (10) -0,0283 | (12) -0,0341 | 0,7971 | всех переменных. Сильно зависит от всех |
| ПМФ | (5) -0,0620 | (2) -0,1113 | (3) -0,2269 | 0,9980 | переменных. Сильно зависит от всех переменных. |
| Байесовский PMF (BPMF) | (4) -0,0628 | (1) -0,1126 | (6) -0,1999 | 0,9817 | Сильно зависит от размера набора данных. |
| Нелинейный PMF (NLPMF) | (6) -0,0611 | (6) -0,0599 | (9) -0,1165 | 0,8786 | Еженедельно зависит от размера набора данных. |
| НПКА | (14) -0,0184 | (11) -0,0213 | (7) -0,1577 | 0,9103 | Сильно зависит от количества элементов. |
| Ранговый CF | (10) -0,0295 | (5) -0,0687 | (10) -0,1065 | 0,8302 | |

Таблица 3: Коэффициенты регрессии для модели $y = м_{ты} з_{ты} + м_{я} з_{я} + м_{г} з_{г} + б$ (1)) где $y$ это МАЭ, $з_{ты}$, $з_{я}$, и $з_{г}$ являются входами от $x_{ты}$, $x_{я}$, и $x_{г}$, нормализованные для достижения схожих масштабов: $з_{ты} = x_{ты} / 10{,}000$, $з_{я} = x_{я} / 3{,}000$, и $з_{г} = x_{г} / 0.05$. Ранг каждой переменной указан в скобках, причем ранг 1 показывает наибольшую зависимость.)

## 3.3 Сравнение точности

Рисунок 5 показывает наиболее эффективный алгоритм (с точки зрения MAE) в зависимости от количества пользователей, количества элементов и плотности. Мы делаем следующие выводы.

1. Выбор наиболее эффективного алгоритма нелинейно зависит от количества пользователей, количества элементов и плотности.

2. NMF доминирует в случаях с низкой плотностью, тогда как BPMF хорошо работает в случаях с высокой плотностью (особенно при большом количестве элементов и пользователей).

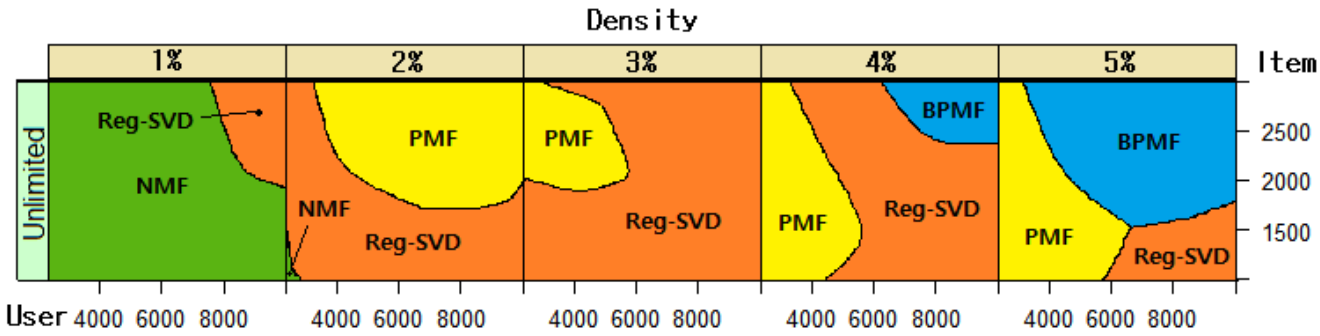


Рисунок 5: Наиболее эффективные алгоритмы в MAE для заданного количества пользователей, количества элементов и плотности.

3. Регуляризированные SVD и PMF хорошо работают при уровнях плотности 2%-4%.

Аналогичные графики RMSE демонстрируют схожие тенденции, при этом регуляризованный SVD превосходит другие алгоритмы в большинстве регионов.

## 3.4 Асимметричные и ранговые метрики

Здесь мы рассматриваем эффект замены MAE или RMSE другими функциями потерь, в частности, асимметричными потерями и потерями на основе ранга.

Асимметричная потеря мотивируется тем фактом, что рекомендация нежелательного элемента хуже, чем избегание рекомендации желаемого элемента. Другими словами, функция потерь $Л(а, б)$, измерение эффекта прогнозирования рейтинга $б$ когда истинный рейтинг $а$ является асимметричной функцией. В частности, мы рассматриваем функцию потерь $Л(а, б)$ определяется следующей матрицей (строки и столбцы выражают количество звезд по шкале от 1 до 5)

$$Л(\cdot, \cdot) = \begin{bmatrix} 0 & 0 & 0 & 7.5 & 10 \\ 0 & 0 & 0 & 4 & 6 \\ 0 & 0 & 0 & 1.5 & 3 \\ 3 & 2 & 1 & 0 & 0 \\ 4 & 3 & 2 & 0 & 0 \end{bmatrix}. \tag{2}$$

Эта функция потерь отражает два убеждения: 1) Разница между рекомендуемыми элементами не важна. Если предположить, что мы выдаем рекомендации с рейтингом 4 или 5, то между ними не засчитывается никакой потери. Точно так же мы не штрафуем за ошибку среди элементов, которые не будут рекомендованы. 2) Мы назначаем более строгий штраф за рекомендацию плохих элементов, чем за пропуск потенциально предпочтительных элементов. В последнем случае потеря — это точная разница между прогнозом и истинной реальностью. Однако в первом случае мы назначаем более высокий штраф. Например, штраф составляет 10 за прогнозирование худшего элемента с истинным баллом 1 как балла 5, что выше 4 для противоположного способа прогнозирования. Во многих практических случаях, связанных с системами рекомендаций, асимметричные функции потерь обеспечивают более реалистичную функцию потерь, чем симметричные функции потерь, такие как MAE или RMSE.

Ранговые функции потерь основаны на оценке ранжированного списка рекомендуемых элементов, представленных пользователю. Оценка списка придает большую важность хорошим рекомендациям в верхней части списка, чем в нижней части списка. Одна конкретная формула, называемая полезностью периода полураспада (HLU) [5, 12], предполагает экспоненциальный спад в позиции списка. Формально, функция полезности, связанная с пользователем $ты$

$$P_{ты} = \sum_{я=1}^{H} \frac{макс(г_{у,я} - г, 0)}{2^{(я-1)/(\alpha-1)}} \tag{3}$$

где $H$ количество рекомендуемых элементов (длина списка), $г_{у,я}$ это рейтинг пользователя $ты$ для товара $я$ в списке, и $г$ и $\alpha$ являются константами, установленными на $г$=3, и $\alpha$= 5 (мы предполагаем, $H$=10). Окончательная функция полезности: $P_{ты}$ деленная на максимально возможную полезность для пользователя, среднее значение по всем пользователям теста [5]. Альтернативные оценки на основе рангов основаны на NDCG [14], тау Кендалла и коэффициенте ранговой корреляции Спирмена [21].

### 3.4.1 Асимметричные потери

На рисунке 6–7 показаны контурные графики равного уровня асимметричной функции потерь (2) как функции количества пользователей, количества элементов и уровня плотности. Мы делаем следующие наблюдения.

1. Форма и плотность расположения контурных линий отличаются от формы контурных линий в случае МАЭ.
2. В целом, регуляризованный SVD превосходит все другие алгоритмы. Другие методы факторизации матриц (PMF, BPMF и NLPMF) работают относительно хорошо для плотных данных. С разреженными данными NMF работает хорошо.

### 3.4.2 Меры оценки на основе рангов

На рисунках 8–9 показаны контуры равного уровня функции HLU (3). На рисунке 10 показан наиболее эффективный алгоритм для разного количества пользователей, количества элементов и плотности. Мы делаем следующие наблюдения.

1. Контурные линии в целом горизонтальны, что указывает на то, что производительность в условиях HLU в значительной степени зависит от количества элементов и в меньшей степени от количества пользователей.
2. Показатель HLU очень чувствителен к плотности набора данных.
3. Регуляризованный SVD превосходит другие методы (см. Рисунок 10) в большинстве настроек. Простое базовое среднее значение лучше всего подходит для небольших и разреженных наборов данных. Аналогичный комментарий можно сделать относительно NPCA. NMF и slope-one хорошо работают для разреженных данных, хотя они несколько отстают от предыдущих упомянутых алгоритмов.

Другие ранговые функции потерь, основанные на NDCG, тау Кендалла и Спирмена, демонстрируют схожие тенденции.

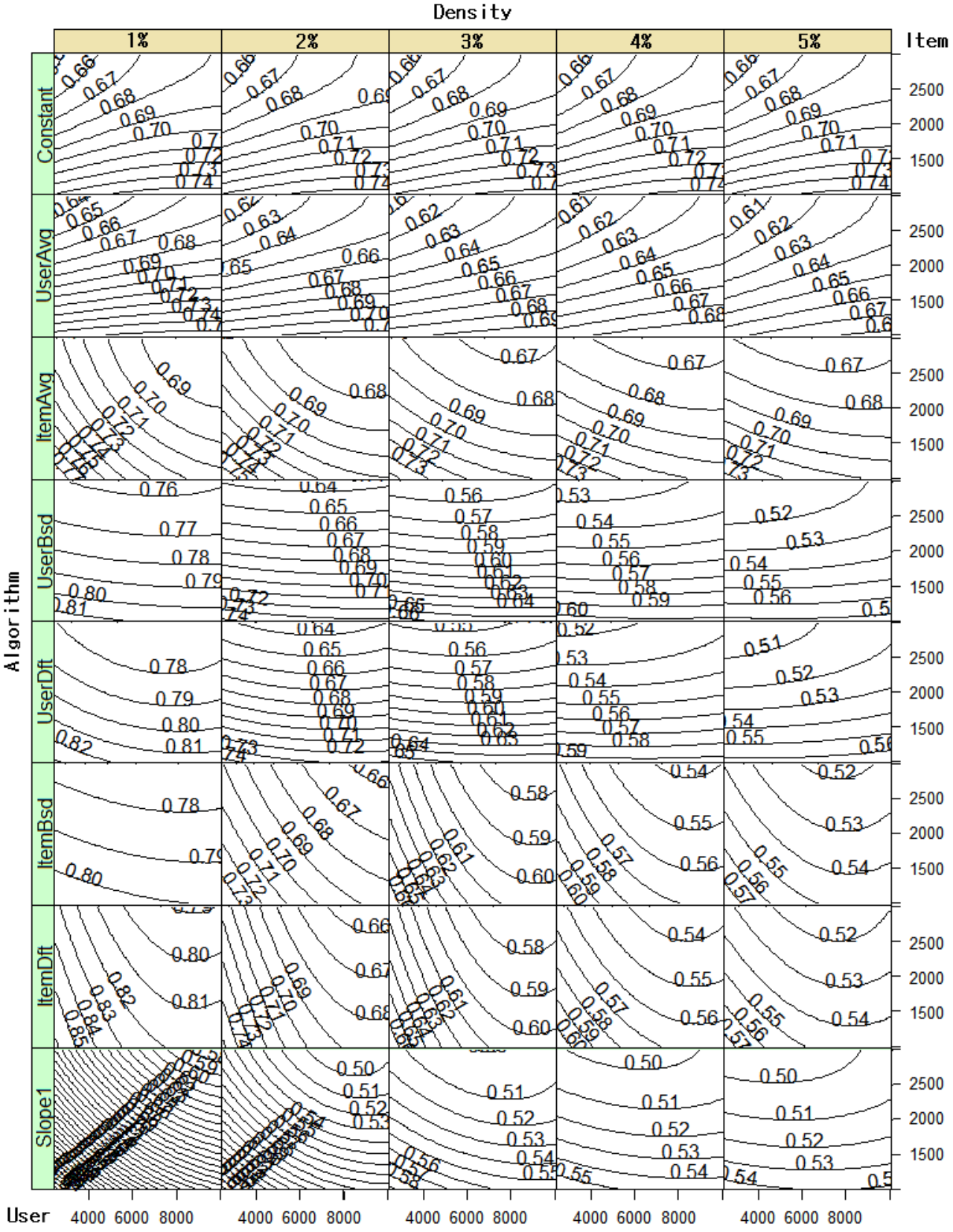


Рисунок 6: Асимметричные контуры потерь для простых методов (более низкие значения означают лучшую производительность.)

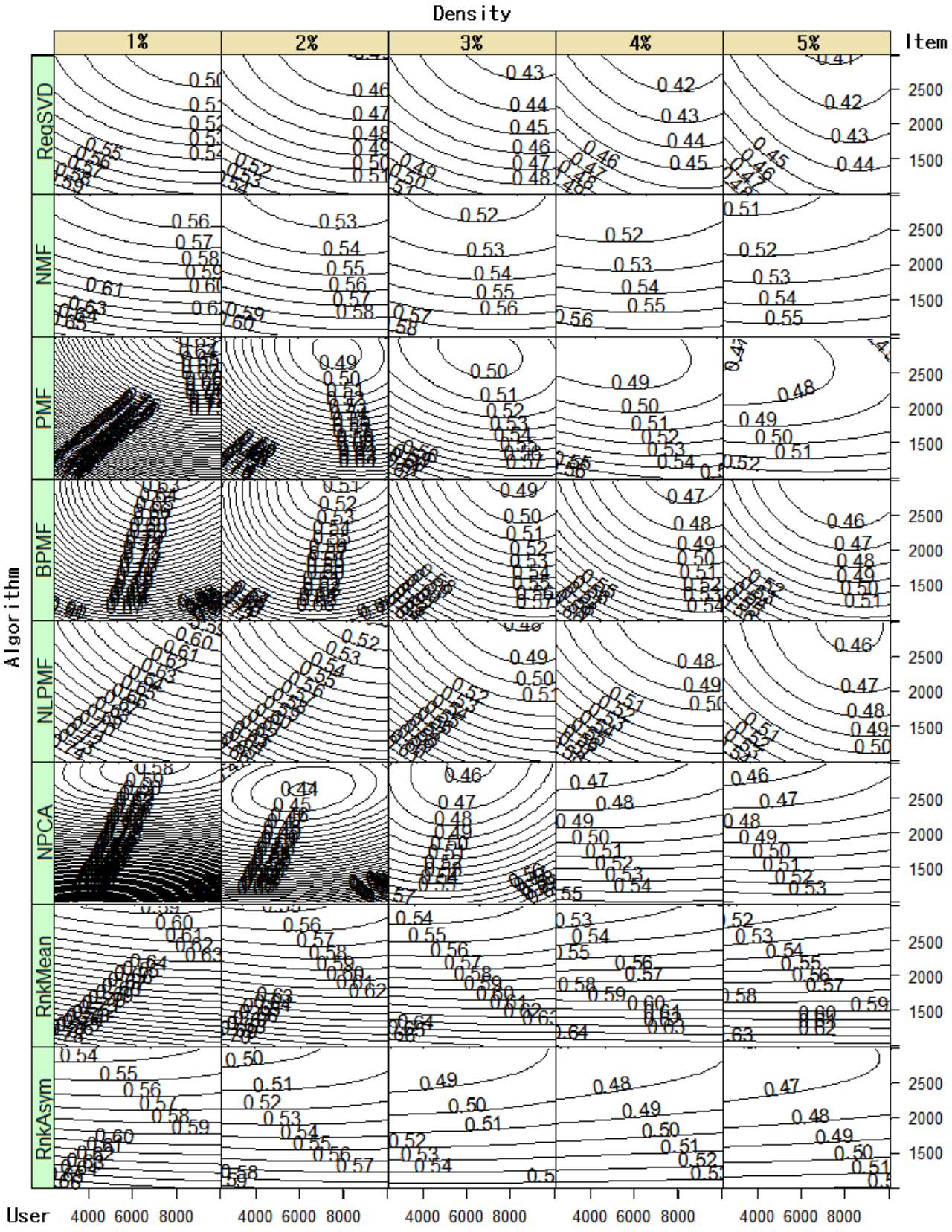


Рисунок 7: Асимметричные контуры потерь для простых методов (более низкие значения означают лучшую производительность.)

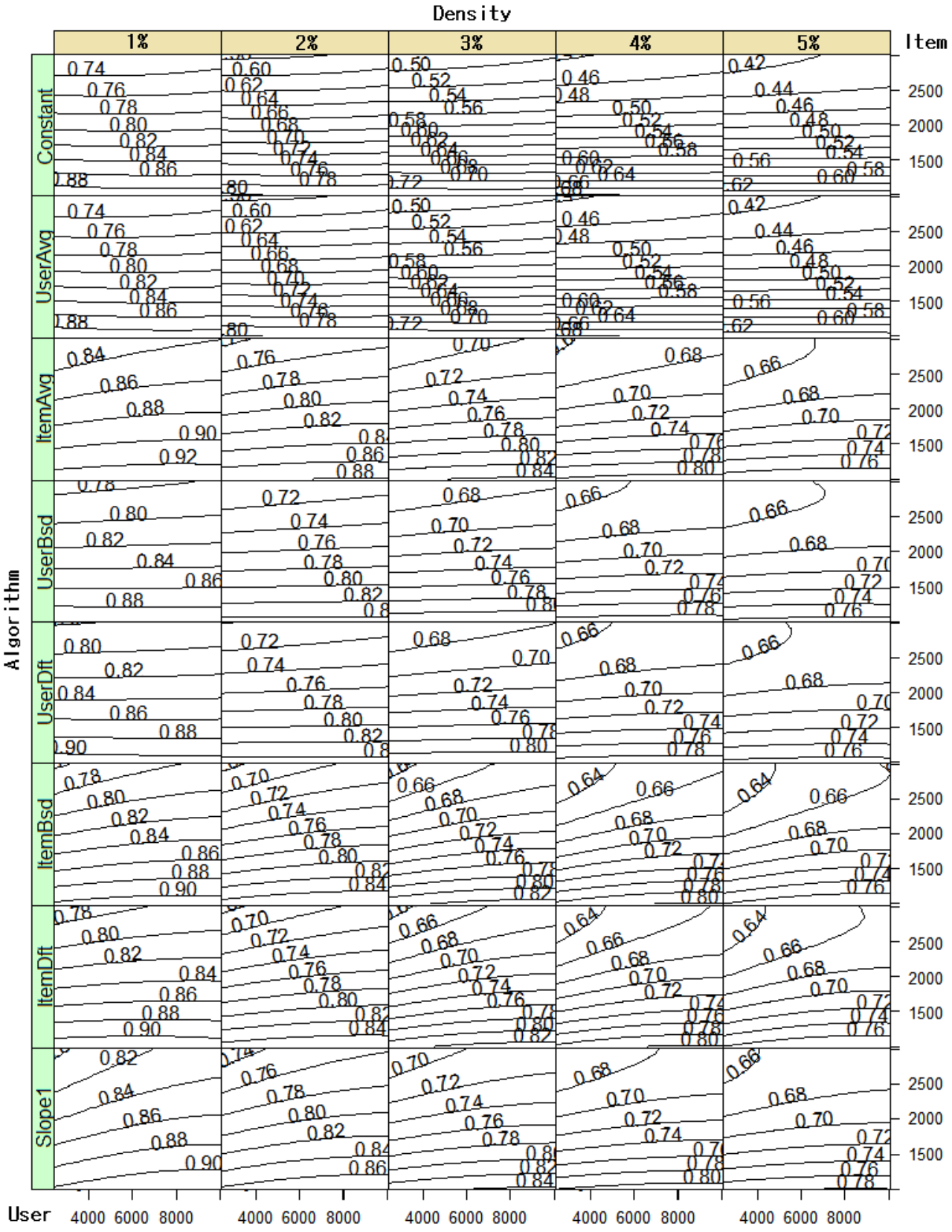


Рисунок 8: Контуры полезности периода полураспада для метода simp1l8e (чем выше значения, тем выше производительность.)

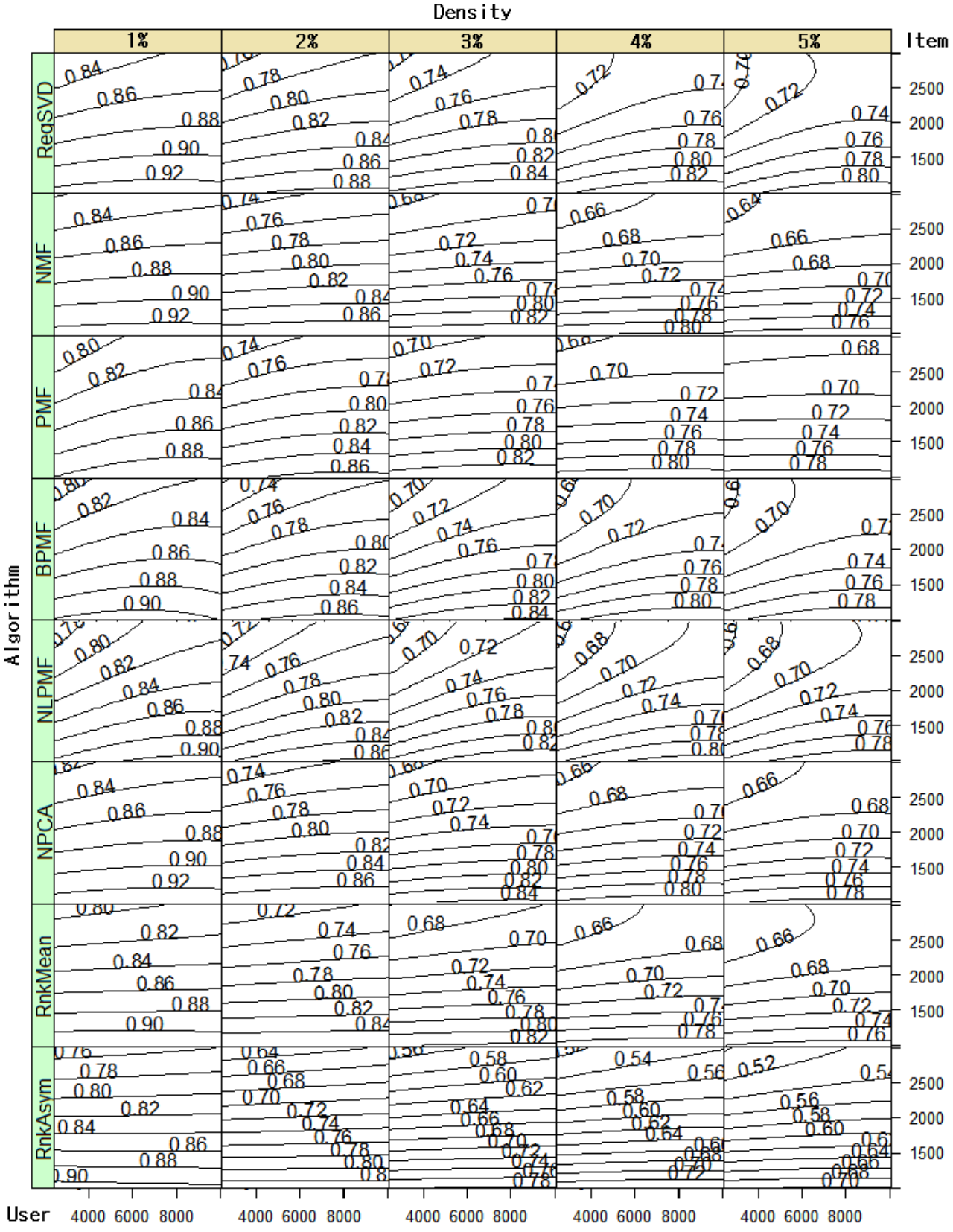


Рисунок 9: Контуры полезности периода полураспада для усовершенствованного метода (чем выше значения, тем выше производительность).

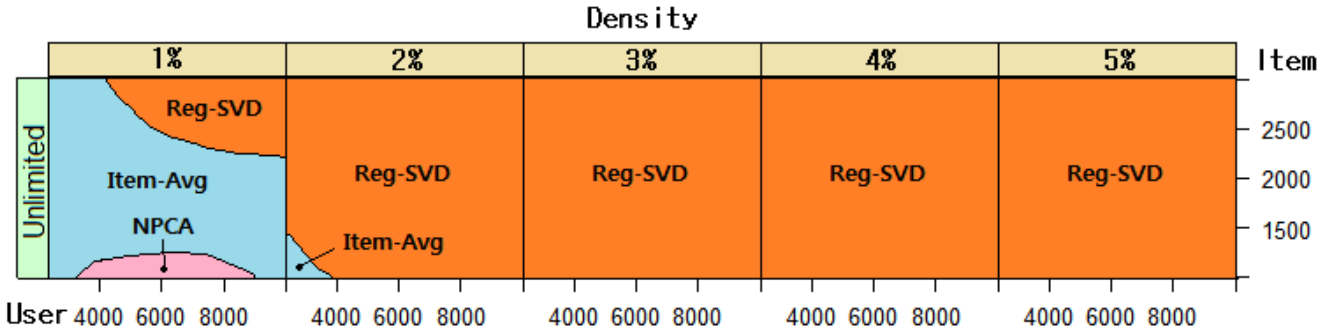


Рисунок 10: Наиболее эффективные алгоритмы в HLU для заданного количества пользователей, количества элементов и плотности.

## 3.5 Вычислительные соображения

Как показано на рисунке 11, время вычисления значительно различается между различными алгоритмами. Поэтому важно учитывать вычислительные проблемы при выборе подходящего алгоритма CF. Мы рассматриваем три различных сценария, перечисленных ниже.

- Неограниченные временные ресурсы:В этом случае мы предполагаем, что можем позволить себе произвольно долгое время вычислений. Этот сценарий реалистичен в некоторых случаях, связанных со статическим обучающим набором, что делает возможным офлайн-вычисление.
- Ограниченный временной ресурс:В этом случае мы предполагаем некоторые умеренные ограничения на время вычислений. Этот сценарий реален в случаях, когда обучающий набор периодически обновляется, что требует периодического повторного обучения с обновленными данными. Мы предполагаем здесь, что фаза обучения должна закончиться в течение часа или около того. Поскольку практические наборы данных, такие как полный набор Netflix, намного больше, чем подвыборочный в наших экспериментах, мы используем гораздо более короткий временной лимит: 5 минут и 1 минуту.
- Приложения реального времени:В этом случае мы предполагаем, что существуют серьезные ограничения на время вычислений. Этот сценарий реален в случаях, когда обучающий набор непрерывно меняется. Мы предполагаем здесь, что фаза обучения не должна превышать нескольких секунд.

На рисунках 5 и 11 показан наиболее эффективный алгоритм CF (с точки зрения MAE) в нескольких различных случаях ограничений по времени в зависимости от количества пользователей, количества элементов и плотности.

Мы делаем следующие замечания.

1. При отсутствии ограничений на вычисления применяются выводы из предыдущих разделов. В частности, NMF лучше всего работает для разреженных наборов данных, BPMF лучше всего работает для плотных наборов данных, а регуляризованный SVD и PMF лучше всего работают в противном случае (PMF хорошо работает с меньшим количеством пользователей, тогда как регуляризованный SVD хорошо работает с меньшим количеством элементов).

2. Когда ограничение по времени составляет 5 минут, Regularized SVD, NLPMF, NPCA и Rankbased CF (те, которые окрашены темным цветом на рисунке 11) не рассматриваются. В этой настройке,

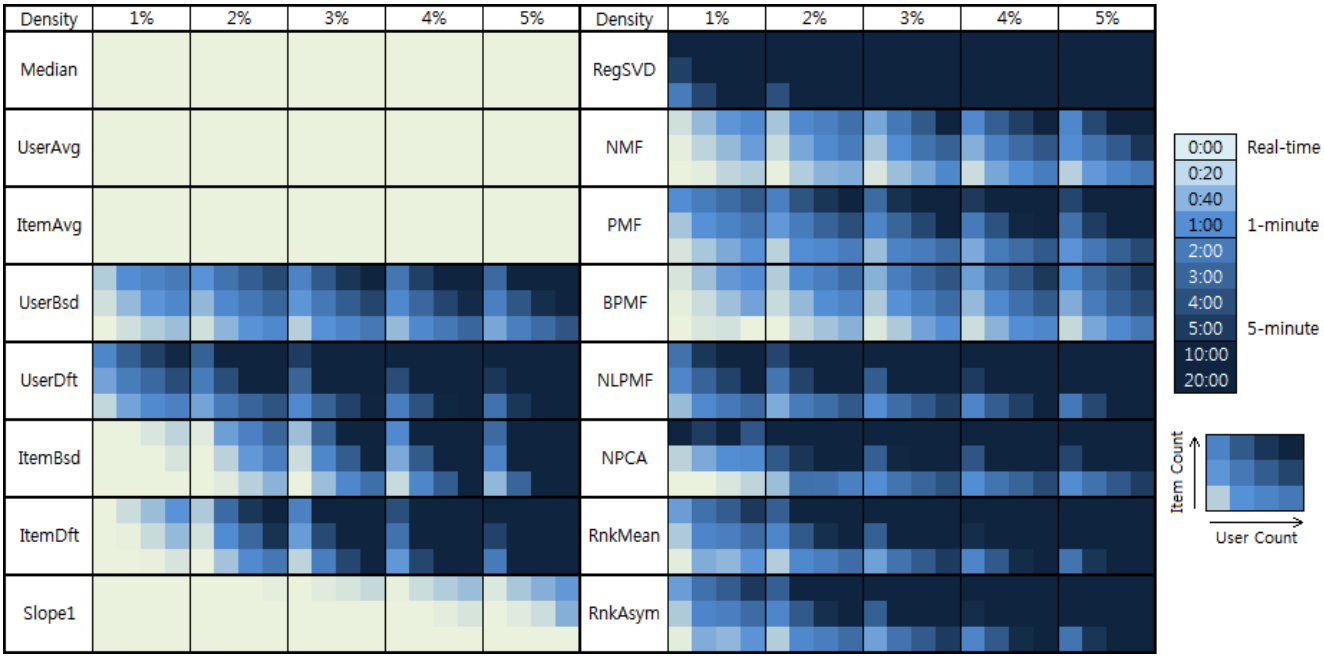


Рисунок 11: Время вычисления (время обучения + время теста) для каждого алгоритма. Легенда справа указывает связь между цветовой схемой и временем вычисления. Ограничения по времени (5 минут и 1 минута), используемые в этой статье, также отмечены. Количество пользователей увеличивается слева направо, а количество элементов увеличивается снизу вверх в каждой ячейке. (То же самое, что и на рисунке 3–4.)

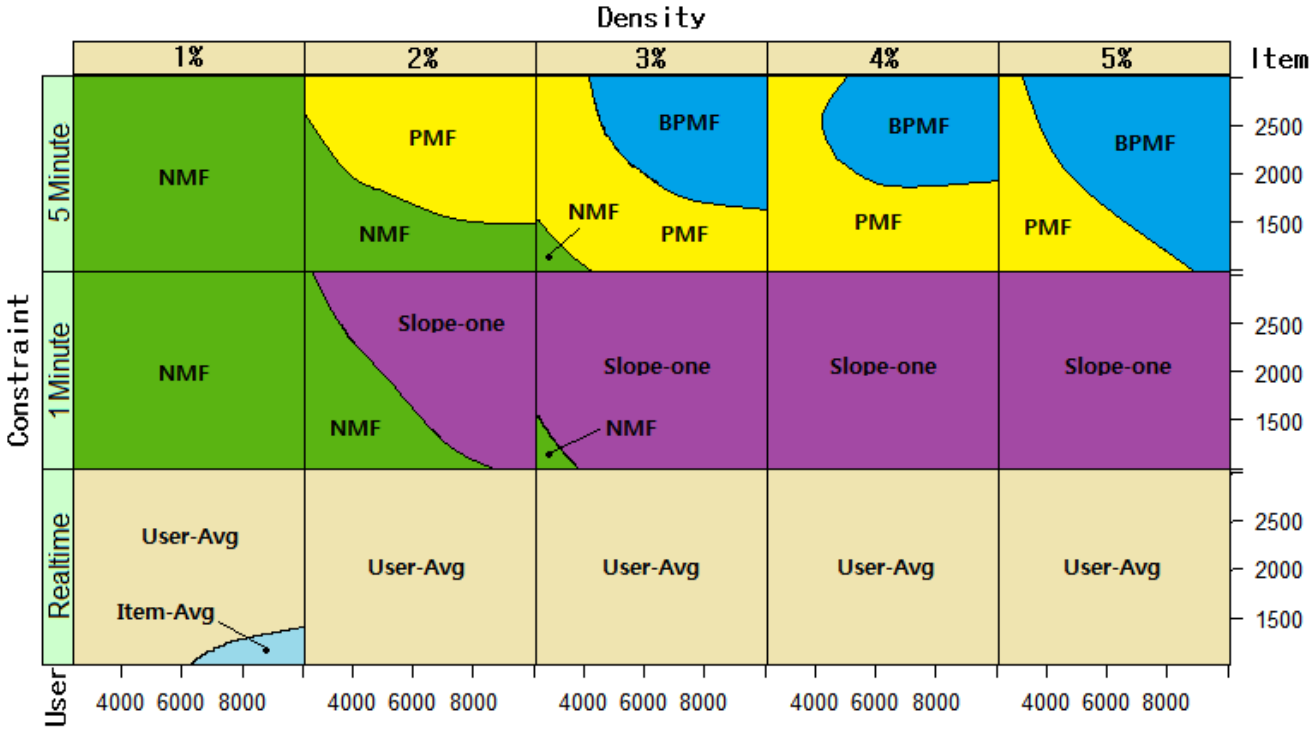


Рисунок 12: Наиболее эффективные алгоритмы с различными ограничениями.

NMF лучше всего подходит для разреженных данных, BPMF лучше всего подходит для плотных и больших данных, а PMF лучше всего подходит для остальных случаев.

3. Когда ограничение по времени составляет 1 минуту, PMF и BPMF дополнительно исключаются из рассмотрения. Slope-one работает лучше всего в большинстве случаев, за исключением самых разреженных данных, где NMF работает лучше всего.

4. В случаях, требующих вычислений в реальном времени, наилучшим алгоритмом является среднее значение пользователя, за исключением небольшого региона, где предпочтительнее среднее значение элемента.

# 4 Обсуждение

В дополнение к выводам, изложенным в разделе 3, мы выделили семь групп методов КФ, где методы КФ в одной группе имеют определенные общие экспериментальные свойства:

- Базовые показатели: Константа, Среднее значение для пользователя, Среднее значение для элемента
- Методы, основанные на памяти: основанные на пользователе, основанные на элементе (со значениями по умолчанию и без них)
- Матричная факторизация I: регуляризованный SVD, PMF, BPMF, NLPMF
- Матричная факторизация II: NMF
- Другие (по отдельности): Slope-one, NPCA, CF на основе ранга.

В таблице 4 для каждой из этих групп показаны зависимость, точность, вычислительные затраты, а также плюсы и минусы.

Ниже мы повторяем некоторые из основных выводов. Более подробную информацию и дополнительные выводы см. в разделе 3.

- Методы, основанные на матричной факторизации, как правило, имеют самую высокую точность. В частности, регуляризованный SVD, PMF и его вариации работают лучше всего в отношении MAE и RMSE, за исключением очень разреженных ситуаций, где NMF работает лучше всего. Методы матричной факторизации также хорошо работают с точки зрения асимметричных затрат и мер оценки на основе ранга. NPCA и CF на основе ранга также хорошо работают в этих случаях. Метод Slopeone работает хорошо и является вычислительно эффективным. Однако методы на основе памяти не имеют особых достоинств, кроме простоты.
- Все алгоритмы различаются по точности в зависимости от количества пользователей, количества элементов и плотности. Однако сила и характер зависимости варьируются от алгоритма к алгоритму, а двумерные отношения изменяются, когда третьей переменной присваиваются разные значения. В общих случаях высокая зависимость от количества пользователей и количества элементов коррелирует с высокой зависимостью от плотности, которая, по-видимому, является более влиятельным фактором.
- Существует компромисс между лучшей точностью и другими факторами, такими как низкая дисперсия точности, вычислительная эффективность, потребление памяти и меньшее количество настраиваемых параметров. То есть, более точные алгоритмы, как правило, сильно зависят от размера и плотности набора данных, имеют большую дисперсию точности, менее эффективны в вычислительном отношении и имеют больше настраиваемых параметров. Тщательное изучение экспериментальных результатов может помочь разрешить этот компромисс способом, который специфичен для конкретной ситуации. Например, когда вычислительная эффективность менее важна, наиболее подходящими являются методы факторизации матриц, а когда вычислительная эффективность важна, лучшим выбором может быть метод slope-one.

Это экспериментальное исследование, сопровождаемое открытым исходным кодом программного обеспечения, которое позволяет воспроизводить наши эксперименты, проливает свет на то, как алгоритмы CF сравниваются друг с другом, и на их зависимость от параметров задачи. Выводы, описанные выше, должны помочь практикам, внедряющим системы рекомендаций, и исследователям, изучающим новые современные методы CF.

# Ссылки

[1] Г. Адомавичус и А. Тужилин. На пути к следующему поколению рекомендательных систем: обзор современного состояния и возможных расширений. *Труды IEEE по знаниям и инжинирингу данных*, 17(6):734–749, 2005.

[2] C. Basu, H. Hirsh и W. Cohen. Рекомендация как классификация: использование социальной и контентной информации в рекомендации. В *Труды Национальной конференции по искусственному интеллекту*, страницы 714–720, 1998.

| Категория | | Базовые данные | На основе памяти | Матричная факторизация | | Другие | | |
|---|---|---|---|---|---|---|---|---|
| Алгоритмы | | Постоянный, Средний пользователь, Средний показатель по предмету | На основе пользователя, На основе элементов, с дефолтом | Рег-СВД, ПМФ, БПМФ, НЛПМФ | НМФ | Склон-один | НПКА | На основе ранга |
| Зависимость | Размер | Очень низкий | Низкий | Высокий | Низкий | Высокий | Очень низкий | Справедливый |
| | Плотность | Очень низкий | Высокий | Очень высокий | Низкий | Очень высокий | Высокий | Справедливый |
| Точность | Плотный | Очень плохо | Хороший | Очень хороший | Хороший | Хороший | Бедный | Справедливый |
| | Редкий | Бедный | Очень плохо | Справедливый | Очень хороший | Бедный | Бедный | Справедливый |
| Асимметричная точность | | Бедный | Справедливый | Очень хороший | Хороший | Хороший | Очень хороший | Очень хороший |
| HLU/NDCG | | Очень плохо | Справедливый | Очень хороший | Справедливый | Хороший | Справедливый | Справедливый |
| Тау/Копейщик Кендалла | | Очень плохо | Справедливый | Хороший | Справедливый | Справедливый | Справедливый | Справедливый |
| Вычисление | Тренироваться | Нет | Нет | Медленный | Быстрый | Очень быстро | Медленный | Нет |
| | Тест | Очень быстро | Очень медленно | Справедливый | Справедливый | Быстрый | Медленный | Очень медленно |
| Потребление памяти | | Низкий | Высокий | Высокий | Высокий | Низкий | Очень высокий | Высокий |
| Регулируемые параметры | | Нет | Немного | Много | Много | Нет | Немного | Немного |
| Общие достоинства | | Вычисление занимает мало времени. | Не нужно тренироваться. | Выполнить лучше всего с высоким данные о плотности. | Выполнить лучше всего с редкими данные. Поезд быстрый. | Выступать хорошо несмотря на короткое время. | Выступать хорошо при использовании асимметричный меры. | Выступать хорошо при использовании асимметричный меры. |
| Общие недостатки | | Точность очень плохо. | Тестирование занимает очень долгое время. Использует много памяти. | Множество параметров должно быть скорректировано. Вычисление занимает много времени. | Много параметры должно быть скорректировано. | Выполнять плохо без большой/плотный набор данных. | Использует чрезвычайно большая память в течение обучение. | Вычисление берет слишком много времени. |

Таблица 4: Сводка плюсов и минусов алгоритмов рекомендаций

[3] Р. Белл, Ю. Корен и К. Волински. Теперь все вместе: взгляд на премию Netflix. *Шанс*, 23(1):24–29, 2010.

[4] Д. Биллсус и М. Дж. Паццани. Изучение совместных информационных фильтров. В *Труды пятнадцатой международной конференции по машинному обучению*, страницы 46–54, 1998.

[5] Дж. Бриз, Д. Хекерман и К. Кади. Эмпирический анализ предиктивных алгоритмов для совместной фильтрации. В *Труды по неопределенности в области искусственного интеллекта*, 1998.

[6] SHS Chee, J. Han и K. Wang. Rectree: эффективный метод совместной фильтрации. В *Конспект лекций по информатике*, страницы 141–151. Springer Verlag, 2001.

[7] М. Коннор и Дж. Херлокер. Кластеризация элементов для совместной фильтрации. В *Труды семинара ACM SIGIR по рекомендательным системам*, 2001.

[8] С. Дебнат, Н. Гангули и П. Митра. Весовые коэффициенты функций в системе рекомендаций на основе контента с использованием анализа социальных сетей. В *Труды 17-й международной конференции по Всемирной паутине*, страницы 1041–1042, 2008.

[9] Д. ДеКост. Совместное предсказание с использованием ансамблей максимальных факторизаций матрицы маржинального отбора. В *Труды 23-й международной конференции по машинному обучению*, страницы 249–256, 2006.

[10] А. Гунавардана и Г. Шани. Обзор метрик оценки точности рекомендательных задач. *J. Mach. Learn. Res.*, 10:2935–2962, 2009.

[11] Y. Hu, Y. Koren и C. Volinsky. Совместная фильтрация для неявных наборов данных обратной связи. В *Труды Восьмой международной конференции IEEE по интеллектуальному анализу данных 2008 г.*, страницы 263–272, 2008.

[12] Z. Huang, D. Zeng и H. Chen. Сравнение алгоритмов рекомендаций совместной фильтрации для электронной коммерции. *Интеллектуальные системы IEEE*, 22:68–78, 2007.

[13] Д. Яннах, М. Занкер, А. Фельферниг и Г. Фридрих. *Рекомендательные системы – Введение*. Кембридж, 2011.

[14] К. Ярвелин и Й. Кекяляйнен. Оценка ИК-методов на основе совокупного выигрыша. *ACM Transactions по информационным системам*, 20(4):422–446, 2002.

[15] Ким Б. М. и Ли К. Вероятностная модель оценки для совместной фильтрации на основе атрибутов элементов. *Труды Международной конференции IEEE/WIC/ACM по веб-аналитике 2004 г.*, страницы 185–191, 2004.

[16] Y. Koren. Факторизация встречает соседство: многогранная модель совместной фильтрации. В *Труды 14-й международной конференции ACM SIGKDD по обнаружению знаний и интеллектуальному анализу данных*, страницы 426–434, 2008.

[17] Н. Д. Лоуренс и Р. Уртасун. Нелинейная матричная факторизация с гауссовыми процессами. В *Труды 26-й ежегодной международной конференции по машинному обучению*, 2009.

[18] Ли Д. и Сын Х. Изучение частей объектов с помощью неотрицательной матричной факторизации. *Природа*, 401:788–791, 1999.

[19] Д. Ли и Х. Сын. Алгоритмы неотрицательной матричной факторизации. В *Достижения в области нейронных систем обработки информации*. Издательство Массачусетского технологического института, 2001.

[20] Д. Лемир и А. Маклахлан. Предикторы наклона один для совместной фильтрации на основе рейтинга в режиме онлайн. *Общество промышленной математики*, 05:471–480, 2005.

[21] Дж. И. Марден. *Анализ и моделирование ранговых данных*. CRC Press, 1996.

[22] П. Мелвилл, Р. Дж. Муни и Р. Нагараджан. Контент-усиленная совместная фильтрация. В *В материалах семинара SIGIR 2001 года по рекомендательным системам*, 2001.

[23] К. Мияхара и М. Дж. Паццани. Совместная фильтрация с простым байесовским классификатором. В *Труды 6-й Тихоокеанской международной конференции по искусственному интеллекту*, страницы 679–689, 2000.

[24] К. Мияхара и М. Дж. Паццани. Улучшение совместной фильтрации с помощью простого байесовского классификатора 1. (11), 2002.

[25] А. Патерек. Улучшение регуляризованного сингулярного разложения для совместной фильтрации. *Статистика*, 2007:2–5, 2007.

[26] DY Павлов и DM Пеннок. Подход максимальной энтропии к совместной фильтрации в динамических, разреженных, многомерных областях. В *Труды по нейронным системам обработки информации*, страницы 1441–1448. MIT Press, 2002.

[27] DM Pennock, E. Horvitz, S. Lawrence и CL Giles. Совместная фильтрация по диагностике личности: гибридный подход на основе памяти и модели. В *Труды 16-й конференции по неопределенности в искусственном интеллекте*, 2000.

[28] Дж. Ренни и Н. Сребро. Быстрая факторизация матрицы максимального запаса для совместного предсказания. В *Труды 22-й международной конференции по машинному обучению*, стр. 719. ACM, 2005.

[29] Салахутдинов Р. и Мних А. Байесовская вероятностная матричная факторизация с использованием метода Монте-Карло цепи Маркова. *Труды Международной конференции по машинному обучению*, 2008.

[30] Салахутдинов Р., Мних А. Вероятностная матричная факторизация. *Достижения в области нейронных систем обработки информации*, 2008.

[31] Б. Сарвар, Г. Карипис, Дж. Констан и Дж. Рейдл. Алгоритмы рекомендаций совместной фильтрации на основе элементов. В *Труды международной конференции по Всемирной паутине*, 2001.

[32] BM Sarwar, G. Karypis, J. Konstan и J. Riedl. Рекомендательные системы для крупномасштабной электронной коммерции: Масштабируемое формирование соседства с использованием кластеризации. В *Труды 5-й Международной конференции по вычислительной технике и информационным технологиям*, 2002.

[33] Дж. Силл, Г. Такач, Л. Макки и Д. Лин. Линейное стекирование с весовыми характеристиками. *Препринт Arxiv arXiv:0911.0460*, 2009.

[34] Н. Сребро, Дж. Д. М. Ренни и Т. С. Яакола. Матричная факторизация с максимальной маржой. В *Достижения в области нейронных систем обработки информации 17*, страницы 1329–1336. MIT Press, 2005.

[35] X. Su, R. Greiner, TM Khoshgoftaar, и X. Zhu. Гибридные коллаборативные алгоритмы фильтрации с использованием смеси экспертов. В *Труды Международной конференции IEEE/WIC/ACM по веб-аналитике*, 2007.

[36] X. Су и Т. М. Хошгофтаар. Обзор методов совместной фильтрации. *Adv. в Искусств. Интел.*, 2009:4:2–4:2, январь 2009 г.

[37] М. Сан, Г. Ливан и П. Кидвелл. Оценка вероятностей в системах рекомендаций. В *Труды Международной конференции по искусственному интеллекту и статистике*, 2011.

[38] Л. Х. Унгар и Д. П. Фостер. Методы кластеризации для совместной фильтрации. В *Семинар AAAI по системам рекомендаций*, 1998.

[39] С. Вучетич и З. Обрадович. Коллаборативная фильтрация с использованием подхода, основанного на регрессии. *Системы знаний и информации*, 7:1–22, 2005.

[40] GR Xue, C. Lin, Q. Yang, WS Xi, HJ Zeng, Y. Yu и Z. Chen. Масштабируемая совместная фильтрация с использованием сглаживания на основе кластера. В *Труды 28-й ежегодной международной конференции ACM SIGIR по исследованиям и разработкам в области информационного поиска*, страницы 114–121. ACM Нью-Йорк, Нью-Йорк, США, 2005 г.

[41] К. Ю, С. Чжу, Дж. Лафферти и И. Гонг. Быстрая непараметрическая матричная факторизация для крупномасштабной совместной фильтрации. В *Труды международной конференции ACM SIGIR по исследованиям и разработкам в области информационного поиска*, 2009.

[42] C.-N. Ziegler, G. Lausen и S.-TL Расчет рекомендаций по продуктам на основе таксономии. *Труды тринадцатой международной конференции ACM по управлению информацией и знаниями*, страницы 406–415, 2004.