

УДК 004.6:004.032.26

ББК 16.26

Л82

Лю Лу, Чжан Юйцзюнь

Л82 Рекомендательные системы на основе больших языковых моделей / пер. с кит. И. Л. Люско. – М.: ДМК Пресс, 2026. – 316 с.: ил.

ISBN 978-5-93700-438-3

В книге представлено глубокое исследование ключевых технологий, практических методов и передовых тенденций рекомендательных систем в эпоху больших моделей. Начиная с основ рекомендательных систем и принципов работы больших моделей, в ней систематически разбираются основные способы интеграции больших моделей с рекомендательными системами. Материал подкрепляется множеством реальных практических кейсов, охватывающих сложные и критически важные сценарии.

Издание предназначено для алгоритмических инженеров, системных архитекторов, исследователей и разработчиков, желающих овладеть технологиями рекомендательных систем следующего поколения.

УДК 004.6:004.032.26

ББК 16.26

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

ISBN (кит.) 978-7-115-67556-9

ISBN (рус.) 978-5-93700-438-3

Copyright © 2019 by Julian Havil

© Оформление, издание, перевод, ДМК Пресс, 2025

Предисловие

Рекомендательные системы как базовая инфраструктура эпохи интернета уже проникли во все виды цифровых платформ и прикладных сценариев. С момента появления концепции в 1992 году, пройдя более чем 30-летнюю эволюцию, рекомендательные системы превратились из теоретических исследований в зрелую технологическую область, сочетающую академическую глубину и коммерческую ценность. В последние годы под двойным воздействием взрывного роста объемов данных и скачкообразного повышения вычислительных мощностей были достигнуты прорывные успехи в технологиях машинного обучения и глубокого обучения, что привело рекомендательные системы к беспрецедентным технологическим преобразованиям. Появление нового поколения технологий искусственного интеллекта, представленного большими языковыми моделями (такими как серия GPT от OpenAI, DeepSeek от DeepSeek и др.), не только перестроило технологическую парадигму рекомендательных систем, но и обеспечило качественный скачок в точности рекомендаций, адаптивности к сценариям и пользовательском опыте взаимодействия.

В традиционном процессе разработки рекомендательных систем команды разработчиков обычно вынуждены вкладывать огромные усилия в привлечение данных и оптимизацию алгоритмов: день за днем анализировать признаки поведения пользователей, пробовать различные модели машинного и глубокого обучения, многократно обучать и подбирать гиперпараметры для повышения эффективности рекомендаций. Такой режим «конструирования признаков + итераций модели» хотя и позволяет в определенной степени поддерживать точность рекомендаций и скорость отклика системы, создавая базовую ценность для бизнеса, однако его ограничения становятся все более очевидными – перед лицом быстро меняющихся бизнес-сценариев и огромных объемов гетерогенных данных инженерам приходится непрерывно искать новые технические решения, но часто они попадают в ловушку убывающей предельной отдачи¹ и не могут добиться качественного прорыва.

Под воздействием технологий больших языковых моделей (LLM) методология и рабочие процессы традиционных рекомендательных систем переживают беспрецедентные вызовы и трансформацию. Концепция «технологического равноправия», представленная DeepSeek, позволяет малым и средним предприятиям получать эффективные возможности вывода и обобщения

¹ «Ловушка убывающей предельной отдачи» в разработке рекомендательных систем возникает, когда дополнительные инвестиции в улучшение модели – увеличение объема данных, усложнение архитектуры, расширение числа кандидатов на этапе отбора или рост вычислительных ресурсов – приводят к все меньшему приросту ключевых метрик (точность, NDCG, полнота охвата, удовлетворенность пользователей). После достижения определенного уровня качества каждая новая оптимизация дает минимальный эффект при значительных затратах, а иногда вызывает переобучение или ухудшение онлайн-результатов. – *Прим. ред.*

с относительно низкими затратами за счет локального развертывания больших моделей¹, тем самым значительно повышая точность и адаптивность рекомендательных систем. Этот тренд не только ускоряет итеративное обновление технологий рекомендаций, но и продвигает общую смену парадигмы в отрасли.

В настоящее время революционные технологии искусственного интеллекта перестраивают все отрасли. Как практикующие специалисты в области рекомендательных систем, мы не только имеем возможность следовать за волной развития технологий больших моделей, но и можем активно участвовать в этой технологической революции, помогая предприятиям снижать затраты и повышать эффективность, а также создавать большую ценность для общества.

Именно поэтому мы начали работу над этой книгой. В ней мы, опираясь на собственный накопленный опыт в области больших языковых моделей и рекомендательных систем, в сочетании с передовыми технологическими прорывами отрасли и реальными примерами внедрения, систематически изложим, как применять технологии больших языковых моделей при проектировании, разработке и реализации рекомендательных систем. Наша цель – с помощью доступного и понятного изложения помочь читателям сформировать целостную когнитивную методологию применения больших языковых моделей для усиления рекомендательных систем, овладеть ключевыми путями технологического усовершенствования и развить способность к предвидению будущих тенденций развития отрасли.

Мы хотели бы выразить особую благодарность родным, друзьям и коллегам за их всестороннюю поддержку в процессе написания книги – именно благодаря вашей бескорыстной помощи мы смогли полностью сосредоточиться на творчестве. Также искренне благодарим редакторскую команду издательства за профессиональное руководство и ценные рекомендации. Наконец, выражаем глубокое уважение пионерам отрасли, продвигающим инновационное слияние больших языковых моделей и рекомендательных систем, – ваши передовые исследования предоставили богатую теоретическую основу и практические ориентиры для этой книги.

Лю Лу, Чжан Юйцзюнь

¹ Если не оговорено особо, под «большими моделями» в настоящей книге понимаются большие языковые модели. – *Прим. авт.*

Введение

В эпоху информационного взрыва нас окружает колоссальный объем товаров, новостей, рекламы и другой информации, из-за чего возможности отдельного пользователя по ее обработке серьезно перегружены. Именно в таких условиях возникли рекомендательные системы и стали неотъемлемой частью интернета.

Рекомендательные системы незаметно работают как в потребительских сервисах (2С) – электронная коммерция, новостные порталы, онлайн-обучение, стриминг аудио и видео, – так и в корпоративных решениях (2В): оптимизация цепочек поставок, маркетинг, управление клиентскими отношениями, подбор персонала. Анализируя поведение и предпочтения пользователей, они позволяют бизнес-системам быстро находить в гигантских массивах данных именно тот контент, который действительно нужен пользователю, существенно повышая качество пользовательского опыта.

1.1. РЕКОМЕНДАТЕЛЬНЫЕ СИСТЕМЫ ВОКРУГ НАС

В условиях информационной перегрузки интернета пользователю крайне сложно самостоятельно отыскать интересный контент – здесь решающую роль играют рекомендательные системы. Их задача – предоставлять персонализированные рекомендации для онлайн-продуктов и сервисов, улучшая пользовательский опыт.

Рекомендательные системы встречаются повсюду. Вот типичные сценарии применения:

- в сфере 2С: рекомендованные плейлисты в музыкальных сервисах, товарные рекомендации в интернет-магазинах, персонализированные новостные ленты, подбор курсов на образовательных платформах;
- в сфере 2В: поиск надежных поставщиков, выявление потенциальных клиентов, оптимизация маркетинговых стратегий, подбор кандидатов.

Основной принцип – использование предпочтений пользователя (user), характеристик элемента¹ (item) и данных о взаимодействиях (покупки, клики

¹ В рекомендательных системах элементы – это объекты, предлагаемые пользователю. Основные виды: товары (e-commerce), медиаконтент (фильмы, музыка, видео, подкасты), новостные статьи, вакансии, пользователи (социальные сети, дейтинг), места и заведения (карты, туризм), образовательные курсы, реклама, игры и приложения. По свойствам элементы делятся на долгоживущие (книги, фильмы) и короткоживущие (новости, тренды), а также холодные (новые без истории) и горячие (популярные). – *Прим. ред.*

и т. д.) для фильтрации и ранжирования огромного количества элементов с целью предложить пользователю те, которые с наибольшей вероятностью ему понравятся. Таким образом, рекомендательная система выступает «мостом» между производителями контента и потребителями. Хотя область применения не ограничена интернетом, именно в информационно насыщенной веб-среде ее ценность проявляется особенно ярко.

Рекомендательная система – мощная и сложная серверная инфраструктура, включающая обработку больших данных, разработку алгоритмов и бизнес-логику. Она моделирует сложные взаимосвязи между пользователями, элементами и их взаимодействиями. Обычно состоит из модулей моделирования признаков пользователей и элементов, этапов грубого отбора (recall) и ранжирования (ranking), а также сложной инженерной практики: архитектурные итерации, обновление моделей в реальном времени и офлайн.

Основные задачи рекомендательных систем:

- снижение информационной перегрузки, помощь в быстром получении ценного или интересного контента;
- глубокое понимание пользователей для предоставления персонализированного контента и услуг;
- обнаружение новых элементов, соответствующих пользовательским предпочтениям, что помогает бизнесу расширять рынки;
- повышение кликабельности, конверсии, удержания пользователей и, как следствие, доходов компании.

Рекомендательные системы – одновременно «двигатель» эффективного доступа пользователей к контенту и «двигатель» достижения коммерческих целей интернет-компаний. Эти две роли взаимно усиливают друг друга, продвигая развитие персонализированных сервисов.

С точки зрения пользователя система анализирует исторические данные и «предугадывает» потенциально интересный контент. Потребности пользователей разнообразны: иногда нужен конкретный товар, но выбор слишком велик; иногда пользователь не может четко сформулировать запрос; иногда он сам точно не знает, что ему нужно. В отличие от поисковых систем, требующих явного запроса, рекомендательные системы работают проактивно, автоматически предлагая контент на основе накопленной истории.

С точки зрения бизнеса рекомендательные системы помогают глубже понимать пользователей, удерживать их в нужный момент, снижать затраты на взаимодействие, повышать опыт, активность и конверсию, создавая новые коммерческие возможности и обеспечивая устойчивый рост. У разных платформ – разные цели оптимизации: видеоплатформы фокусируются на длительности просмотра и количестве воспроизведений, электронная коммерция – на покупках и выручке, новостные платформы – на репостах и комментариях.

Персонализация – ключевая особенность, требующая богатых данных о поведении пользователя. Развитие больших данных и ИИ радикально ускорило персонализированные рекомендации. Пример с новостями: от традиционных порталов (Sina News) к приложениям вроде Toutiao, которые формируют полностью персонализированную ленту на главной странице. Все современные ре-

комендательные системы строятся вокруг триады «пользователь – контекст – элемент», независимо от вертикали (новости, видео, e-commerce, реклама).

В будущем развитие больших моделей откроет для рекомендательных систем новые горизонты: более глубокое понимание описаний объектов, учет долгосрочных и динамических предпочтений, точное сопоставление признаков объектов и запросов пользователей, что сделает рекомендации интеллектуальнее и создаст дополнительную бизнес-ценность.

1.2. ЭВОЛЮЦИЯ ТЕХНОЛОГИЙ РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМ

По количеству обслуживаемых пользователей различают персонализированные и групповые рекомендательные системы. Персонализированные предсказывают следующий объект для конкретного пользователя – именно они наиболее глубоко исследованы и являются основной темой книги. Основные подходы: коллаборативная фильтрация, контентные и гибридные методы, среди которых коллаборативная фильтрация остается самой распространенной.

- 1992 – Белкин и Крофт показали, что рекомендательные системы основаны на информационной фильтрации [1]. Вскоре Голдберг с соавт. при разработке Tapestry ввели термин «коллаборативная фильтрация» (Collaborative Filtering, CF) [2]. В последующие годы CF стала доминирующей технологией в новостях, музыке, видео, e-commerce и кино.
- 2006–2009 – конкурс Netflix Prize привлек внимание к матричным разложениям. Работа Копен с соавт. (2009) систематизировала подход и изменила стандарты оценки [3].
- 2007 – логистическая регрессия¹ (LR) для предсказания CTR² в рекламе [4].
- 2010 – факторизационные машины (FM) [5], 2016 – FFM с учетом полей признаков³ (Field-aware Factorization Machines) [6].
- 2014 – Facebook предложил гибридный GBDT + LR [8].

¹ Логистическая регрессия – это статистический метод машинного обучения для задач бинарной классификации. Модель предсказывает вероятность принадлежности объекта к положительному классу (от 0 до 1) с помощью сигмоидной функции. Обучается путем максимизации логарифмического правдоподобия (или минимизации кросс-энтропии) с помощью градиентного спуска. Несмотря на название «регрессия», это классификатор. Простая, интерпретируемая, хорошо работает на линейно разделимых данных и часто используется как базовая модель. – *Прим. ред.*

² CTR (Click-Through Rate, или коэффициент кликабельности) – это показатель, который измеряет процент пользователей, кликнувших на определенный элемент (например, рекламу или ссылку), относительно общего числа пользователей, которые его увидели. В контексте искусственного интеллекта CTR часто используется для оценки эффективности рекламных кампаний, рекомендационных систем и других сервисов, зависящих от пользовательского взаимодействия. – *Прим. ред.*

³ FFM (Field-aware Factorization Machines) – это расширение классической модели FM, которое специально предназначено для задач с категориальными признаками, разбитыми на разные «поля» (fields), например пользователь, товар, категория, бренд, время суток и т. д. В отличие от обычных FM, где все признаки взаимодействуют через одну общую матрицу скрытых факторов, в FFM каждый признак из одного поля имеет отдельный набор скрытых векторов для каждого другого поля. Это позволяет модели более точно захватывать специфические взаимодействия между разными типами признаков. – *Прим. ред.*

- 2017 – Alibaba представила смешанную логистическую регрессию (MLR, Mixed Logistic Regression) [7].
- После триумфа AlphaGo начался бум глубоких нейронных сетей (Deep Neural Networks, DNN) в рекомендательных системах. YouTube существенно повысил точность видеорекомендаций с помощью DNN [9]; технологию быстро подхватили Baidu, Alibaba, Tencent и др.

В области рекомендательных систем одним из важнейших применений глубокого обучения стали модели векторных представлений, или эмбединги (embedding). Они способны преобразовывать дискретную информацию в непрерывные числовые представления. Технология векторных представлений изначально возникла в обработке естественного языка; фундамент для нее заложил алгоритм word2vec, предложенный Миколовым с соавт. [10]. Благодаря своей простоте и высокой эффективности векторные представления очень быстро были перенесены в рекомендательные системы.

Применение глубокого обучения (Deep Learning, DL) в рекомендательных системах в основном развивалось вокруг двух ключевых направлений: автоматизации конструирования признаков и повышения способности моделировать пересечения признаков. Эта эволюция прошла через ряд знаковых моделей – от DSSM [11], Wide&Deep [12] и DeepFM [13] до более поздних решений. Начиная с 2018 года в рекомендательные системы активно внедряется механизм внимания (attention mechanism), что позволило значительно точнее моделировать предпочтения пользователей. SASRec [14] – одна из первых последовательностных рекомендательных моделей, полностью построенная на механизме самовнимания (self-attention); она способна одновременно улавливать долгосрочные и краткосрочные предпочтения пользователя из истории его поведения. Alibaba последовательно предложила целую линейку моделей на основе внимания: DIN [15] (Deep Interest Network), DIEN [16] (Deep Interest Evolution Network), MIMN [17] (Multi-channel Interest Memory Network), DSIN [18] (Deep Session Interest Network).

Все они используют механизмы внимания для отслеживания эволюции интересов пользователя во времени. Ранние рекомендательные модели в основном опирались на коллаборативную фильтрацию¹ и матричную факторизацию² (Matrix Factorization, MF) – они были сравнительно простыми и содержали ограниченное число параметров. С появлением глубоких нейронных сетей выразительная способность моделей резко возросла, а производительность существенно улучшилась. В частности, последовательные модели на основе внимания способны выделять ключевую информацию из длинных цепочек взаимодействий пользователя с элементами, что заметно повышает точность рекомендательных систем. В последние годы, с раз-

¹ Коллаборативная фильтрация – класс методов рекомендаций, которые предсказывают интерес пользователя к элементу, опираясь только на историю взаимодействий множества пользователей и элементов (матрица «пользователь–элемент»), без использования содержательных признаков контента. – *Прим. ред.*

² Матричная факторизация – семейство методов совместной фильтрации, которые раскладывают разреженную матрицу «пользователь × элемент» (размером $m \times n$) на произведение двух матриц низкого ранга. – *Прим. ред.*

витиём больших мультимодальных моделей (Large-Scale Models, LSM), появилась возможность решать множество рекомендательных задач внутри одной единой модели и гибко адаптировать ее под различные даунстрим-сценарии. Это кардинально меняет традиционные бизнес-процессы построения рекомендательных систем.

1.3. ОБЗОР БОЛЬШИХ МОДЕЛЕЙ

Большие языковые модели (LLM, Large Language Models), или просто большие модели, – языковые модели на базе глубоких нейронных сетей с десятками и сотнями миллиардов параметров, обученные методом самообучения на огромных объемах размеченного текста [19].

После появления трансформеров¹ (Transformer) [20] развитие ускорилося. К 2024 году OpenAI, Google, Meta, Baidu, iFlyTek, Alibaba, Huawei и многие другие выпустили десятки мощных моделей, показывающих выдающиеся результаты в задачах обработки естественного языка. Релиз ChatGPT в ноябре 2022 года стал поворотным моментом, продемонстрировав, что одна модель (GPT-3.5/4) способна решать задачи понимания и генерации текста, включая программирование, перевод и диалог.

В отличие от традиционных моделей глубокого обучения с умеренным числом параметров, большие модели содержат от десятков миллионов до триллионов параметров. Чем больше масштаб, тем выше обобщающая способность. При достижении критического размера возникают «эмерджентные способности» (emergent abilities) [21] – новые сложные свойства, которые невозможно предсказать по поведению малых моделей.

Большие модели не только демонстрируют владение знаниями о мире и глубокое понимание языка, но и обладают большей выразительной силой по сравнению с традиционными моделями машинного обучения и глубокого обучения. Они могут объединять знания о мире для выявления потенциальных закономерностей и ключевой информации в данных, а также адаптироваться к различным задачам и бизнес-потребностям на основе бизнес-сценариев и характеристик данных.

1.4. ИНТЕГРАЦИЯ БОЛЬШИХ МОДЕЛЕЙ И РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМ

Возникновение больших моделей (LLM) принесло революционные изменения в обработку естественного языка, и индустрия активно использует их возможности для значительного повышения эффективности рекомендательных

¹ Трансформер – это архитектура нейронной сети, разработанная для обработки последовательностей данных, таких как текст. Она основана на механизме внимания, который позволяет модели фокусироваться на разных частях входной последовательности при генерации выходных данных. Трансформеры способны параллельно обрабатывать данные, что значительно ускоряет обучение и улучшает качество работы с длинными последовательностями. Эта архитектура стала основой для многих современных моделей в области обработки естественного языка, таких как BERT и GPT. – *Прим. ред.*

систем [22]. По сравнению с традиционными рекомендательными системами ключевое преимущество интеграции больших моделей заключается в следующем: они способны извлекать высококачественные векторные представления текстовых признаков и обладают огромным объемом знаний о мире, накопленным в процессе предобучения [23].

Большие модели мастерски улавливают контекстную информацию, что позволяет им глубже понимать текстовые данные пользователей (запросы, отзывы на заказы, описания товаров и т. д.) [24]. В результате существенно повышается точность рекомендательных систем и удовлетворенность пользователей.

Кроме того, способности больших моделей к нулевому обучению¹ и обучение на нескольких примерах² по промптам (zero-shot learning и few-shot learning) [25] открывают эффективный путь решения проблемы холодного старта³ в условиях разреженных исторических взаимодействий, создавая новые практические возможности для рекомендательных систем.

Выдающаяся обобщающая способность больших моделей позволяет рекомендовать ранее не встречавшиеся элементы-кандидаты. Это достигается благодаря тому, что в процессе предобучения модель накопила огромное количество фактических знаний, профессиональных сведений в предметной области (доменных сведений) и здравого смысла. Даже столкнувшись с совершенно новым пользователем или новым товаром, она способна выдавать обоснованные и качественные рекомендации.

В настоящее время интеграция больших моделей и рекомендательных систем стала одним из важнейших исследовательских направлений как в академической, так и в среде разработчиков. Большие модели находят широкое применение на всех этапах рекомендательной воронки: от извлечения (retrieval) и ранжирования (ranking) до конструирования признаков⁴ и рекомендаций при холодном старте. При этом в процессе практического внедрения больших моделей в рекомендательные системы по-прежнему сохраняется ряд крити-

¹ Нулевое обучение – модель решает задачу без единого примера из этой задачи. Ей дается только описание задачи на естественном языке (промпт). Пример: «Переведи на французский: ...» – и модель переводит, хотя ее никогда специально не настраивали на перевод. – *Прим. ред.*

² Обучение на нескольких примерах – модели дают очень мало примеров (обычно 1–64) прямо в промпте, и она сразу умеет решать задачу. Пример: в промпте 5 примеров «кошка → котенок, собака → щенок», дальше «медведь → ?», модель отвечает «медвежонок». – *Прим. ред.*

³ Проблема холодного старта (cold start problem) в рекомендательных системах – это ситуация, когда модель не может дать точные рекомендации новому пользователю (нет истории взаимодействий) или новому элементу (нет оценок/просмотров от пользователей). Кратко: отсутствие данных о «холодном» объекте приводит к низкому качеству рекомендаций на старте; решается через контентные признаки, популярные объекты или гибридные подходы. – *Прим. ред.*

⁴ Конструирование признаков – это процесс выбора, преобразования и создания новых признаков (или «фич»), которые используются для обучения моделей машинного обучения. Основная цель конструирования признаков заключается в улучшении производительности модели, обеспечивая ей более информативные и релевантные данные. – *Прим. ред.*

чески важных вызовов: интерпретируемость результатов, обеспечение справедливости¹ (fairness), защита конфиденциальности пользователей и другие этические и технические проблемы.

СПИСОК ЛИТЕРАТУРЫ

- [1] Belkin N.J., Croft W.B. Information filtering and information retrieval: two sides of the same coin? [J]. Communications of the ACM, 1992, 35 (12): 29–38.
- [2] Goldberg D., et al. Using collaborative filtering to weave an information TAP-ESTRY [J]. Communications of the ACM, 1992, 35 (12): 61–70.
- [3] Koren Y., Bell R., Volinsky C. Matrix Factorization Techniques for Recommender Systems [J]. Computer, 2009, 42 (8): 30–37.
- [4] Richardson M., et al. Predicting Clicks: Estimating the Click-Through Rate for New Ads [C] // WWW2007, 2007.
- [5] Rendle S. Factorization Machines [J]. IEEE, 2010.
- [6] Juan Y., et al. Field-aware Factorization Machines for CTR Prediction [C] // RecSys, 2016.
- [7] Gai K., et al. Learning Piece-wise Linear Models from Large Scale Data for Ad Click Prediction [J]. 2017.
- [8] He X., et al. Practical Lessons from Predicting Clicks on Ads at Facebook [M]. 2014.
- [9] Covington P., et al. Deep Neural Networks for YouTube Recommendations [C] // RecSys, 2016.
- [10] Mikolov T., et al. Efficient Estimation of Word Representations in Vector Space [J]. 2013.
- [11] Huang P.S., et al. Learning deep structured semantic models for web search using clickthrough data [C] // CIKM, 2013.
- [12] Cheng H.T., et al. Wide & Deep Learning for Recommender Systems [J]. ACM, 2016.
- [13] Guo H., et al. DeepFM: An End-to-End Wide & Deep Learning Framework for CTR Prediction.
- [14] Kang W.C., Mcauley J. Self-Attentive Sequential Recommendation [C] // ICDM, 2018.
- [15] Zhou G., et al. Deep Interest Network for Click-Through Rate Prediction [J]. 2017.
- [16] Zhou G., et al. Deep Interest Evolution Network for Click-Through Rate Prediction [C] // AAAI, 2019.
- [17] Pi Q., et al. Practice on Long Sequential User Behavior Modeling for Click-Through Rate Prediction [J]. ACM, 2019.

¹ Обеспечение справедливости (fairness) в контексте ИИ относится к процессам и методам, направленным на предотвращение предвзятости и дискриминации в алгоритмах и моделях машинного обучения. Это включает в себя разработку и тестирование систем, которые обеспечивают равное обращение с различными группами пользователей независимо от их расы, пола, возраста или других характеристик. Главное внимание уделяется тому, чтобы результаты, выдаваемые ИИ, были объективными, прозрачными и не приводили к угнетению или стигматизации определенных категорий людей. – Прим. ред.

- [18] Feng Y., et al. Deep Session Interest Network for Click-Through Rate Prediction [J]. 2019.
- [19] Zhao W.X., et al. A Survey of Large Language Models [J]. ArXiv, 2023, abs/2303.18223.
- [20] Vaswani A., et al. Attention is All you Need [C], 2017.
- [21] Wei J., et al. Emergent Abilities of Large Language Models [J]. ArXiv, 2022, abs/2206.07682.
- [22] Chen J. A Survey on Large Language Models for Personalized and Explainable Recommendations [J]. ArXiv, 2023, abs/2311.12338.
- [23] Liu P., et al. Pre-train, Prompt, and Recommendation: A Comprehensive Survey [J]. TACL, 2023, 11: 1553–1571.
- [24] Geng S., et al. Recommendation as Language Processing (RLP): A Unified Pre-train, Personalized Prompt & Predict Paradigm (P5) [C] // RecSys, 2022.
- [25] Sileo D., et al. Zero-Shot Recommendation as Language Modeling [C], 2021.

Основы рекомендательных систем

Рекомендательная система представляет собой сложную систему, состоящую из множества взаимодействующих модулей. При ее проектировании необходимо учитывать выбор алгоритмов, архитектуру системы, обработку информации о пользователях и объектах рекомендаций, а также многие другие аспекты.

Ключевой технологией рекомендательных систем являются рекомендательные алгоритмы, которые на основе поведения и предпочтений пользователя прогнозируют контент, потенциально интересный ему. Тем не менее в реальных сценариях применения такие системы сталкиваются с многочисленными ограничениями и вызовами. Полноценная рекомендательная система должна решать широкий спектр задач: сбор и обработка данных, выбор и обработка признаков, разработка стратегии отбора кандидатов, обеспечение достаточно малого времени отклика для оперативности в реальном времени, решение проблемы холодного старта и т. д. Только эффективное решение всех этих задач позволяет рекомендательной системе полноценно выполнять свои функции.

Рекомендательные системы – область одновременно сложная и насыщенная вызовами, содержащая множество нерешенных проблем. Цель настоящей главы – предоставить руководство по проектированию, разработке, реализации и оценке персонализированных рекомендательных систем, а также освещение всех аспектов, которые необходимо учитывать в процессе разработки. Для удобства изложения сначала будут рассмотрены рекомендательные алгоритмы, а затем инженерные вопросы разработки и реализации при построении систем.

2.1. ОБЗОР БАЗОВЫХ РЕКОМЕНДАТЕЛЬНЫХ АЛГОРИТМОВ

Рекомендательный алгоритм включает два ключевых этапа: вычисление сходства¹ (similarity) между пользователями и элементами, а также ранжирование

¹ В рекомендательных системах термин «сходство» (similarity) обозначает количественную меру близости или степени схожести между объектами – пользователями и элементами (items) по их поведению, предпочтениям или характеристикам в признаковом или латентном пространстве. Оно вычисляется с помощью специальных метрик, таких как косинусное сходство, корреляция Пирсона, коэффициент Джаккарда, инвертированное евклидово расстояние или с скорректированным косинусное сходство (adjusted cosine). В коллаборативной фильтрации сходство используется

по этому сходству для формирования рекомендаций. Существует множество подходов к классификации персонализированных рекомендательных систем; каждый алгоритм обладает своими особенностями и подходит для разных сценариев. В данном разделе основное внимание уделяется методам на основе коллаборативной фильтрации (CF), методам на основе признаков и методам на основе последовательностей – между ними прослеживается определенная эволюционная преемственность, и они отражают основные этапы развития рекомендательных алгоритмов.

Методы на основе коллаборативной фильтрации – одни из самых классических. К ним относятся коллаборативная фильтрация по элементам (Item-based Collaborative Filtering, ItemCF) и коллаборативная фильтрация по пользователям (User-based Collaborative Filtering, UserCF). Благодаря простоте и прямому использованию поведения и предпочтений пользователей эти методы получили широкое распространение, однако им присущи выраженный эффект популярных объектов¹ (head effect), слабая способность работать с разреженными данными и ограниченная обобщающая способность. Матричная факторизация ввела понятие латентных векторов² для использования в матрице совпадений (co-occurrence matrix), улучшив обработку разреженных матриц и частично устранив ограничения классической коллаборативной фильтрации.

Методы на основе признаков обучают сложную математическую модель для предсказания предпочтений пользователя и вероятности клика (Click-Through Rate, CTR). Они используют признаки пользователей и объектов, прогнозируют рейтинг, а затем на основе актуальной информации о пользователе рекомендуют объекты с наивысшим предсказанным рейтингом. Такие методы позволяют включать в модель дополнительные данные (профиль пользователя, атрибуты элемента), применять отбор и кросс-признаки³, существенно повышая точность персонализации. Модели делятся на две крупные группы: модели на основе классического машинного обучения (LR, FM, GBDT и др.)

по-разному: в user-based подходе – для оценки сходства между пользователями (похожие пользователи одинаково оценивают элементы), в item-based – для сходства между элементами (похожие элементы нравятся одним и тем же пользователям), что позволяет генерировать рекомендации на основе этих мер близости. – *Прим. ред.*

- ¹ Эффект популярных объектов (head effect) относится к ситуации, когда модели, особенно в области рекомендаций или поиска, склонны выдавать результаты, которые подчеркивают наиболее популярные или часто встречающиеся элементы. Это может приводить к недостаточному разнообразию в рекомендациях, так как менее известные, но потенциально интересные элементы получают меньше внимания. Эффект может возникать из-за доминирования популярных элементов в тренировочных данных, что делает модель менее чувствительной к менее распространенным вариантам. – *Прим. ред.*
- ² Латентный вектор (скрытый фактор) – это представление данных в виде вектора в пространстве меньшей размерности, служит абстракцией для описания скрытых или неявных факторов, которые влияют на наблюдаемые данные. В GAN или вариационных автокодировщиках латентный вектор может использоваться для генерации новых данных, которые похожи на обучающие. – *Прим. ред.*
- ³ Кросс-признаки (cross-features), также известные как признаки взаимодействия (interaction features), – это синтетические признаки, создаваемые путем комбинирования двух или более исходных (категориальных или числовых) признаков для улавливания их совместного влияния на целевую переменную. – *Прим. ред.*

и модели на основе глубокого обучения (DSSM¹, Wide&Deep, DeepFM и др.). С ростом числа и размерности признаков именно модели глубокого обучения стали незаменимыми в современных рекомендательных системах.

Вышеперечисленные методы в основном опираются на исторические данные и статические признаки, часто игнорируя последовательность поведения пользователя. В отличие от них методы на основе последовательностей анализируют порядок взаимодействий пользователя с элементами, выявляют динамику изменения интересов и прогнозируют будущее поведение пользователя.

Далее эти три семейства методов будут рассмотрены подробно.

2.2. МЕТОДЫ НА ОСНОВЕ КОЛЛАБОРАТИВНОЙ ФИЛЬТРАЦИИ

Методы на основе коллаборативной фильтрации – классическая технология рекомендаций, полностью опирающаяся на данные взаимодействий пользователей с элементами. Они способны улавливать паттерны поведения и предпочтений пользователя, эффективно работать с большими объемами данных и потому играют важную роль в персонализированных рекомендациях. На ранних этапах построения рекомендательной системы, когда еще отсутствует глубокий анализ бизнеса и развитое конструирование признаков, коллаборативная фильтрация часто становится методом выбора.

2.2.1. Коллаборативная фильтрация

Коллаборативная фильтрация использует исторические данные о поведении пользователя (просмотренные страницы, купленные товары, прослушанные треки и т. д.). Отзывы пользователя делятся на явные (explicit feedback – оценки, рейтинги) и неявные (implicit feedback – клики, просмотры). Основные алгоритмы – ItemCF и UserCF.

ItemCF реализует принцип «элементы, понравившиеся одному пользователю, схожи» (items liked by the same user are similar): если пользователю нравятся определенные элементы, ему будут интересны похожие на них. Алгоритм вычисляет сходство между всеми элементами по степени предпочтения пользователей, а затем для каждого любимого элемента целевого пользователя находит n наиболее похожих и рекомендует их.

UserCF реализует принцип «пользователи с похожими вкусами любят похожие элементы» (users with similar taste like similar items): целевому пользователю рекомендуются элементы, которые нравятся пользователям, наиболее близким к нему по вкусу. Вычисляется сходство между пользователями, выбираются k самых похожих, и рекомендуются элементы, которые они предпочли.

¹ DSSM (Deep Structured Semantic Model) – это модель глубокого обучения, разработанная для обработки и извлечения семантической информации из текстовых данных. Она используется в таких задачах, как поиск информации, рекомендации и обработка естественного языка. Основная идея DSSM заключается в том, чтобы представлять текстовые объекты (например, документы и запросы) в виде векторов в высокоразмерном семантическом пространстве, что позволяет вычислять их семантическую схожесть и улучшать качество поиска. – *Прим. ред.*

В качестве примера рассмотрим систему рекомендации фильмов: после получения всех предпочтений пользователей по фильмам оценки пользователей и фильмов можно преобразовать в матрицу совместных появлений¹ (co-occurrence matrix). На рис. 2.1 показана матрица совместных появлений. В матрице слева строки представляют пользователей, столбцы – фильмы, а числа в матрице – оценки пользователя (от 1 до 5) соответствующего фильма. Символ «?» означает, что пользователь не смотрел фильм. На практике матрицы совместных появлений, как правило, разрежены, и для заполнения пропущенных значений обычно используется импутация, например позиции, соответствующие символу «?», заполняются нулями.

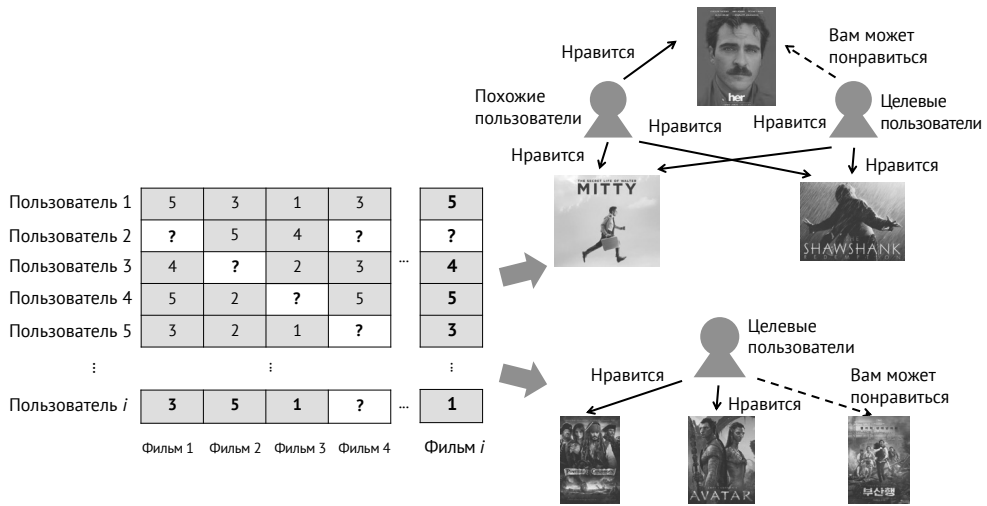


Рис. 2.1. Матрица совместных появлений пользователь–фильм

Вычисление сходства является ключевым элементом алгоритмов коллаборативной фильтрации. Среди наиболее распространенных метрик сходства:

- косинусное сходство (cosine similarity);
- коэффициент корреляции Пирсона (Pearson correlation coefficient);
- евклидово расстояние (Euclidean distance);
- манхэттенское расстояние (Manhattan distance) и ряд других.

В данном разделе подробное рассмотрение алгоритмов коллаборативной фильтрации будет проведено на примере именно косинусного сходства.

(1) Расчет сходства в UserCF

Для пользователей A и B сходство вычисляется по формуле косинусного сходства их векторов признаков \mathbf{a} и \mathbf{b} :

$$\text{sim}_{\text{user}}(A, B) = \cos(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \cdot \|\mathbf{b}\|}$$

¹ Также используются варианты термина – «матрица совпадений» и «матрица совместных появлений». – Прим. перев.

Здесь \mathbf{a} и \mathbf{b} представляют собой векторы признаков пользователей A и B соответственно.

После вычисления сходства пользователей и нахождения k пользователей, наиболее похожих на целевого, можно спрогнозировать предпочтения целевого пользователя на основе известных оценок этих похожих пользователей. Прогнозируемые предпочтения целевого пользователя в отношении элементов получаются путем вычисления средневзвешенного значения сходства пользователей и уровней предпочтений (рейтингов) похожих пользователей.

(2) Расчет сходства в ItemCF

Формула расчета сходства для элементов C и D выглядит аналогично:

$$\text{sim}_{\text{item}}(C, D) = \cos(\mathbf{c}, \mathbf{d}) = \frac{\mathbf{c} \cdot \mathbf{d}}{\|\mathbf{c}\| \cdot \|\mathbf{d}\|},$$

где \mathbf{c} и \mathbf{d} представляют собой векторы признаков элементов C и D соответственно.

После вычисления степени сходства элементов и нахождения n наиболее похожих элементов на целевой элемент предпочтения пользователя можно спрогнозировать на основе известных оценок этих похожих элементов. Прогнозируемое предпочтение пользователя целевому элементу определяется путем вычисления средневзвешенного значения степени сходства элементов и оценок пользователей.

Формула для расчета предпочтений пользователя выглядит следующим образом:

$$P = \sum(\text{sim} \cdot \text{score}).$$

UserCF и ItemCF имеют разные сильные стороны, выбор зависит от сценария. При росте числа пользователей и элементов матрица совместных появлений становится огромной, резко возрастает вычислительная сложность. Кроме того, сохраняются проблемы: выраженный эффект популярных элементов, слабая обобщающая способность, проблема холодного старта и др.

2.2.2. Матричная факторизация

Матричная факторизация (Matrix Factorization, MF) [1] была предложена для устранения проблем выраженного эффекта популярных элементов и слабой обобщающей способности классической коллаборативной фильтрации. MF раскладывает матрицу совместных появлений (матрицу совпадений пользователь–элемент) на две низкоранговые матрицы, получая латентные векторы пользователей и элементов (рис. 2.2). Скалярное произведение этих векторов отражает степень предпочтения.

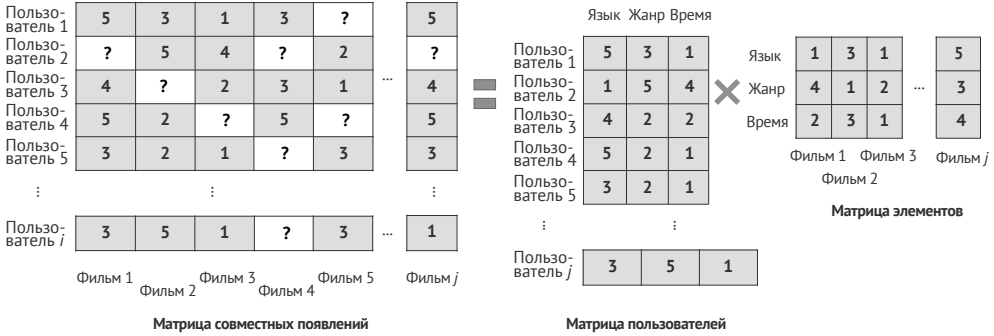


Рис. 2.2. Процесс матричной факторизации

В практических алгоритмах рекомендательных систем процесс прогнозирования предпочтений целевых пользователей посредством матричной факторизации включает вычисление произведения векторов латентных факторов пользователей и элементов. Если число пользователей равно m , а число элементов равно n , то матрица совместных появлений пользователей и элементов M имеет размер $m \times n$. После матричной факторизации мы получаем матрицу пользователей U и матрицу элементов V . Матричная факторизация может быть выражена в виде произведения следующим образом:

$$M_{m \times n} \approx U_{m \times k} V_{n \times k}^T$$

Здесь k – размерность вектора скрытых факторов, указывающая на то, что предпочтения пользователя в отношении элемента в основном определяются k факторами. В рекомендательной системе фильмов k может быть языком фильма, временем, жанром и т. д. (как показано на рис. 2.3). Значение k необходимо устанавливать в соответствии с фактической ситуацией, чтобы сбалансировать эффективность рекомендации и вычислительные затраты. К распространенным методам расширения матричной факторизации относятся базовая матричная факторизация (Basic Matrix Factorization), регуляризованная матричная факторизация (Regularized Matrix Factorization), базовое сингулярное разложение (Basic Singular Value Decomposition, SVD) и расширенное сингулярное разложение (Extended Singular Value Decomposition – ESVD или SVD++).

Цель матричной факторизации – максимально приблизить произведение матрицы M пользователя U и матрицы элементов V , полученной в результате факторизации матрицы M , к истинным предпочтениям пользователя, сохраняя при этом как можно больше исходной информации из матрицы совместных появлений, тем самым достигая прогнозирования предпочтений пользователя. Целевая функция матричной факторизации может быть выражена как

$$\min_{U, V} \sum_{(u,k) \in S} (U_u V_k^T - \text{score}(u, k))^2$$

Здесь S представляет выборку предпочтений всех пользователей относительно элементов, т. е. оценки всех пользователей. Функция $\text{score}(u, k)$ представляет оценку пользователя u для элемента k .



Рис. 2.3. Матричная факторизация матрицы совместных появлений пользователь–фильм

Хотя матричная факторизация преодолевает некоторые ограничения коллаборативной фильтрации, по сути она представляет собой оптимизацию классических методов ItemCF и UserCF без изменения их формы. Следовательно, она не приводит к значительному повышению производительности рекомендательных систем. Матричная факторизация по-прежнему сталкивается с такими проблемами, как проблема холодного старта, разреженность данных, плохая масштабируемость и вычислительная сложность.

2.2.3. Ограниченные машины Больцмана

Ограниченная машина Больцмана (Restricted Boltzmann Machine, RBM) [2] – это генеративная стохастическая нейронная сеть. В 2007 году Джеффри Хинтон, отец глубокого обучения, и другие ученые выступили с предложением применения RBM в глубоком обучении на Международной конференции по машинному обучению (ICML). С тех пор RBM использовалась для решения задачи коллаборативной фильтрации в конкурсе Netflix Prize, продемонстрировав свой потенциал в области рекомендательных систем.

RBM состоит из двух слоев: первый называется видимым, а второй – скрытым. Связь между этими двумя слоями ненаправленная, а единицы измерения в каждом слое представляют собой двоичные переменные. Каждый узел в видимом слое представляет собой низкоуровневый признак элемента в наборе данных. RBM точно настраивают параметры модели, изучая распределение выборок данных, чтобы более точно отражать предпочтения пользователя в отношении элементов. Для каждого пользователя обучается отдельная RBM, вход – вектор оценок элементов. После обучения веса позволяют восстанавливать пропущенные оценки и прогнозировать предпочтения.

В процессе обучения каждому пользователю соответствует RBM-модель, входным вектором которой являются оценки элементов. Входной вектор каждого пользователя рассматривается как отдельная выборка, и обучение выполняется последовательно. После обучения всех выборок формируются окончательные веса RBM-модели. Используя эти обученные веса, RBM-модель может реконструировать оценки элементов, заданные пользователем, заполнить пропущенные значения (элементы без оценок) и, таким образом, предсказать предпочтения пользователя в отношении этих элементов.

Глубокие сети доверия (Deep Belief Network, DBN) и глубокие машины Больцмана (Deep Boltzmann Machine, DBM) [3] представляют собой более сложные модели, построенные путем последовательного стэкинга (stacking) нескольких ограниченных машин Больцмана (RBM). Основное различие между ними заключается в следующем:

- DBN сочетает свойства направленных и ненаправленных графов;
- DBM является полностью ненаправленной моделью, в которой каждая пара соседних слоев образует отдельную RBM.

Несмотря на теоретическую выразительность, как DBN, так и DBM обладают чрезмерно высокой алгоритмической сложностью, что приводит к низкой скорости обучения и инференса¹. По сравнению с современными моделями глубокого обучения, основанными на обратном распространении ошибки², эти архитектуры не демонстрируют значительных практических преимуществ и в настоящее время практически не используются в промышленных рекомендательных системах.

Хотя эти модели редко используются в промышленных системах из-за сложности, они сыграли важную роль в истории применения глубокого обучения к рекомендательным системам.

¹ Инференс (от англ. inference) – процесс получения предсказаний (выхода) из обученной модели по новым входным данным. В контексте генеративных моделей (DBN, DBM, RBM и т. п.) также используется термин «вывод» (или «вывод вероятностей»), но в последние годы в русскоязычных текстах по глубокому обучению и большим моделям «инференс» полностью прижился и стал стандартным. – *Прим. ред.*

² Обратное распространение ошибки (backpropagation) – основной алгоритм обучения многослойных нейронных сетей. Он состоит из двух проходов: прямого прохода (forward pass) – входные данные проходят через сеть, вычисляются активации и итоговое предсказание, и обратного прохода (backward pass) – вычисляется ошибка на выходе, затем с помощью правила цепочки (chain rule) градиенты ошибки последовательно «распространяются» назад от выходного слоя к входному, слой за слоем. На каждом слое градиенты используются для обновления весов по методу градиентного спуска. Именно благодаря обратному распространению ошибки стало возможным эффективное обучение глубоких нейронных сетей с десятками и сотнями слоев.

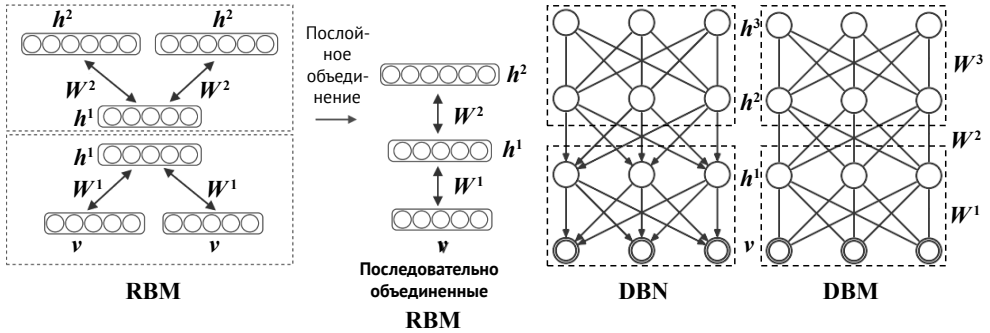


Рис. 2.4. Структура моделей RBM, DBN (направленный граф), DBM (ненаправленный граф)

2.2.4. Коллаборативная фильтрация с глубоким обучением

После появления алгоритмов глубокого обучения традиционные методы коллаборативной фильтрации¹ получили новое развитие. Основная идея заключается в применении аппарата глубокого обучения для получения представлений матрицы совместных появлений «пользователь–элемент». Главное преимущество такого подхода – возможность интегрировать вспомогательную информацию (side information) и использовать ее в процессе предсказания, что заметно повышает качество рекомендаций.

Модель глубокой коллаборативной фильтрации

Модель глубокой коллаборативной фильтрации (Deep Collaborative Filtering, DCF) [4] объединяет подходы глубокого обучения с классической коллаборативной фильтрацией на основе матричной факторизации. В качестве входных данных модель принимает матрицу совместных появлений пользователь–элемент, а также векторы признаков пользователей X и элементов Y . DCF относится к гибридным архитектурам и реализует тесную интеграцию матричной факторизации с изучением признаков.

Структура модели DCF представлена на рис. 2.5. Модель обучает скрытые факторы пользователей U и элементов V , используя следующие источники информации: матрицу взаимодействий R , векторы признаков пользователей X и векторы признаков элементов Y . Ключевая особенность – применение маргинализованных шумоподавляющих автокодировщиков (marginalized denoising autoencoder) для кодирования признаков пользователей и элементов в низкоразмерные скрытые представления². Полученные скрытые факторы затем используются для восстановления и предсказания элементов матри-

¹ Встречается также вариант перевода «совместная фильтрация». – Прим. перев.

² Низкоразмерные скрытые представления (low-dimensional hidden representations) – это компактные форматы представления данных, которые сохраняют важные характеристики или семантику исходных данных, но занимают меньше пространства и имеют меньшую размерность. Такие представления часто используются в машинном обучении и глубоком обучении для упрощения анализа и обработки данных, а также для улучшения производительности моделей. – Прим. ред.

цы R , что обеспечивает высокоточное прогнозирование оценок и формирование персонализированных рекомендаций.

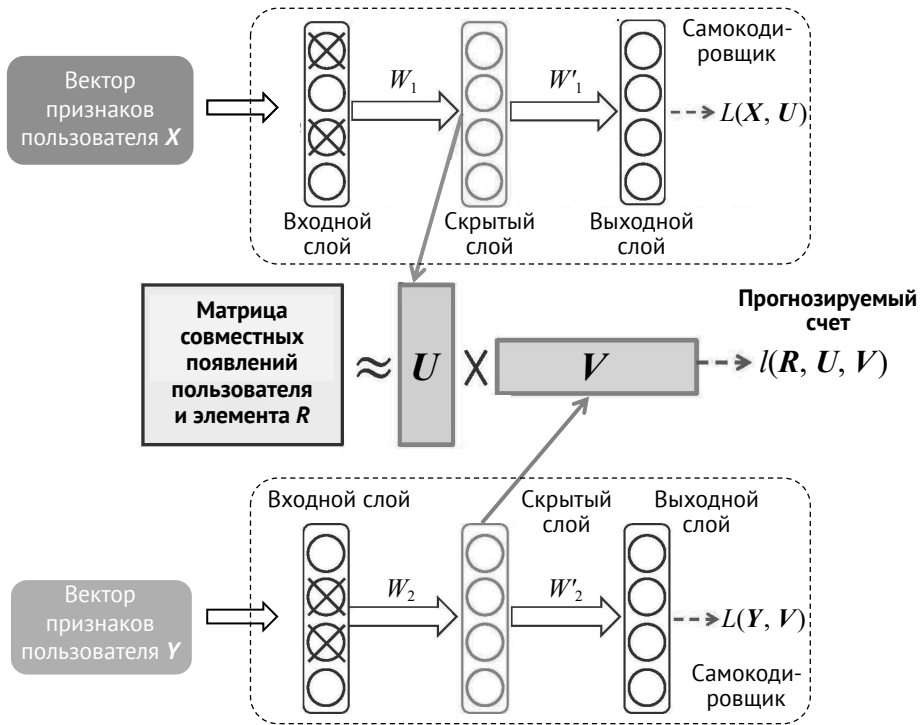


Рис. 2.5. Структура модели DCF

Модель глубокой матричной факторизации

Модель глубокой матричной факторизации (deep matrix factorization, DMF) [5] представляет собой архитектуру матричной факторизации, построенную на основе нейронных сетей. DMF отображает информацию о взаимодействиях пользователей и элементов в низкоразмерное векторное пространство. В отличие от классической матричной факторизации, DMF использует нелинейные многослойные перцептроны¹ (multi-layer perceptron, MLP), что значительно повышает выразительную способность модели. Структура модели DMF показана на рис. 2.6. Векторы пользователей и векторы элементов подаются в две независимые нейронные сети для извлечения признаков. На выходе этих сетей формируются латентные представления пользователей и элементов соответственно. Далее предсказание оценки пользователя для элемента выполняется путем вычисления сходства (обычно косинусного) между полученными латентными векторами.

¹ Многослойный перцептрон – это тип искусственной нейронной сети, состоящий из нескольких слоев нейронов, который используется для решения задач классификации и регрессии. – Прим. ред.

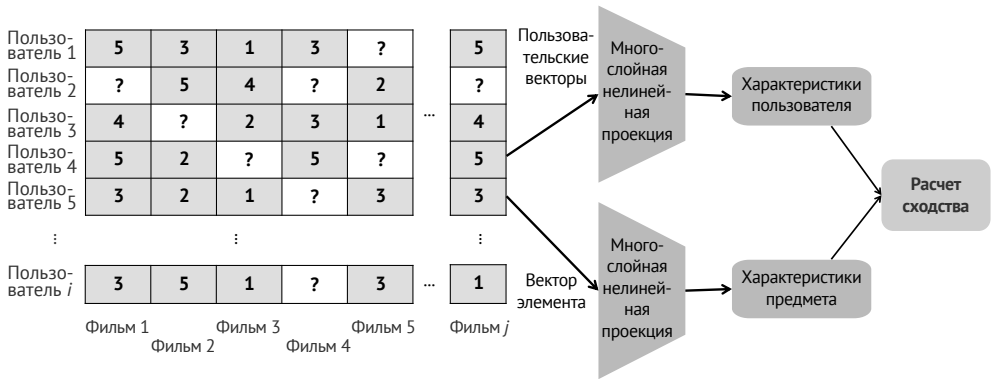


Рис. 2.6. Структура модели DMF

Модель DMF сохраняет исходные оценки пользователей из матрицы совместных появлений (например, шкала от 1 до 5 баллов) и не использует дополнительную вспомогательную информацию для описания пользователей и элементов. Векторы пользователей и элементов подаются в отдельные нейронные сети, которые проецируют их в одно и то же низкоразмерное пространство. Предсказание оценки выполняется через вычисление скалярного произведения полученных латентных представлений. За счет применения нелинейных преобразований DMF обеспечивает существенно более сильную выразительную мощност¹ по сравнению с классической матричной факторизацией, сохраняя при этом простоту и высокую эффективность.

Алгоритмы коллаборативной фильтрации получили широкое распространение в области рекомендаций и в определенной степени повысили точность выбора и удовлетворенность пользователей. Тем не менее им присущ ряд фундаментальных ограничений:

- чрезмерная зависимость от исторических взаимодействий и, как следствие, потеря ценной информации из признаков пользователей и элементов;
- склонность к чрезмерному представлению популярных элементов (head items) и игнорированию длинного хвоста;
- выраженная проблема холодного старта при появлении новых пользователей или элементов;
- высокие требования к вычислительным ресурсам и объему памяти при росте объемов данных;
- неспособность адаптироваться к эволюции интересов пользователя и к рекомендации разнообразного контента.

¹ Выразительная мощност (expressive power) модели в машинном обучении – это ее способность аппроксимировать сложные функции и зависимости в данных, то есть насколько богато и гибко модель может представлять разнообразные паттерны без переобучения. Модели с большей выразительной мощностю (например, глубокие нейронные сети или нелинейные трансформации) способны улавливать тонкие нелинейные взаимосвязи, которые простые модели (линейная регрессия, классическая матричная факторизация) не могут, но при этом требуют больше данных и вычислительных ресурсов, чтобы избежать переобучения. – *Прим. ред.*

2.3. МЕТОДЫ РЕКОМЕНДАЦИЙ НА ОСНОВЕ ПРИЗНАКОВ

Как традиционная коллаборативная фильтрация, так и матричная факторизация используют только данные о взаимодействиях, полностью игнорируя богатую информацию, содержащуюся в признаках пользователей и элементов. Это приводит к значительным потерям полезного сигнала. Для преодоления данного недостатка были разработаны методы рекомендаций, основанные на признаках. Такие подходы, опираясь на методы машинного обучения, включая глубокое обучение, позволяют эффективно извлекать и моделировать признаки, включая сложные взаимодействия между ними, что существенно повышает точность распознавания предпочтений пользователей.

2.3.1. Методы на основе машинного обучения

В эпоху больших данных машинное обучение играет ключевую роль в прогнозировании пользовательских предпочтений. Благодаря богатым профилям пользователей и поведенческим данным модели машинного обучения способны строить значительно более точные модели поведения пользователей и рекомендаций товаров, обеспечивая настоящую персонализацию. Методы на основе машинного обучения стали важным шагом вперед по сравнению с традиционной коллаборативной фильтрацией: они учитывают не только взаимодействия «пользователь–элемент», но и контекстные признаки пользователей (местоположение, пол, возраст, образование) и элементов (категория, описание товара и т. д.), что позволяет существенно повысить точность предсказания интересов пользователей.

Логистическая регрессия

Логистическая регрессия [6] (Logistic Regression, LR) – один из базовых и наиболее широко применяемых алгоритмов машинного обучения. В рекомендательных системах она преобразует задачу рекомендации в задачу бинарной классификации¹, где положительными примерами выступают клики пользователя по товарам или просмотры видео.

На рис. 2.7 показан принцип работы алгоритма логистической регрессии (LR). LR обучается на значимых признаках, таких как история взаимодействия пользователя с элементами, описания пользователей и атрибуты элементов, и преобразует эти признаки в числовые входные векторы признаков x . Цель логистической регрессии – найти оптимальные параметры w , минимизирующие расхождение между прогнозами модели и фактическими результатами. В процессе онлайн-прогнозирования LR сортирует рекомендуемые элементы на основе их прогнозируемых значений вероятности, формирует список рекомендаций и рекомендует пользователю N лучших элементов. Математическая формула алгоритма выглядит следующим образом:

¹ Задача бинарной классификации – это задача машинного обучения, где цель состоит в том, чтобы отнести объекты (например, данные) к одному из двух классов. Например, можно классифицировать электронные письма как «спам» или «не спам», а также определить, есть у пациента болезнь или нет (да/нет). Модели для бинарной классификации обучаются на размеченных данных, а затем используются для предсказания класса новых, неразмеченных объектов. – *Прим. ред.*

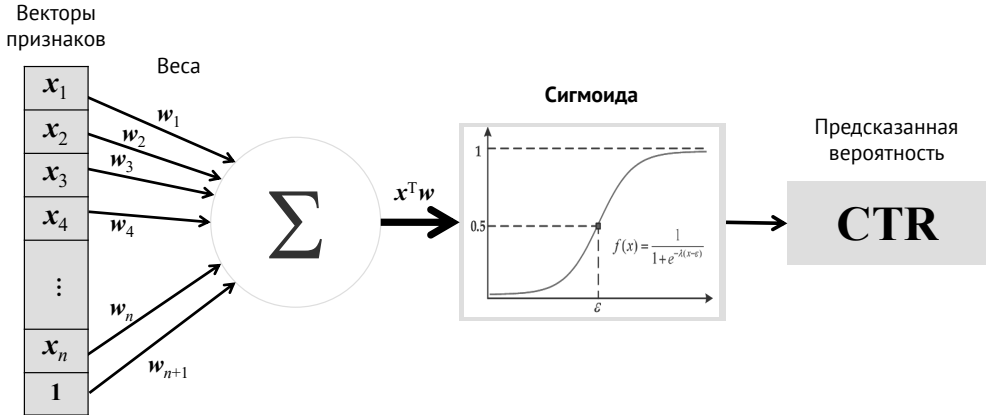


Рис. 2.7. Принципиальная схема алгоритма логистической регрессии

$$\hat{y}(\mathbf{x}) = \frac{1}{1 + e^{-(\mathbf{x}^T \mathbf{w} + b)}}$$

Логистическая регрессия проста в реализации и эксплуатации, однако обладает ограниченной выразительной мощностью при работе со сложными зависимостями и большим числом признаков. Она сильно зависит от ручной разработки признаков, требует значительных трудозатрат и способна моделировать только взаимодействия первого порядка, что приводит к потере важной информации о пересечениях признаков и снижению точности рекомендаций.

Факторизационные машины

Для устранения недостатка обобщенных линейных моделей (включая LR) в моделировании пересечений признаков исследователи сначала предложили метод POLY2 [7]. Он реализует все парные комбинации признаков второго порядка, однако при этом количество весовых параметров возрастает с n до n^2 , что резко увеличивает вычислительную нагрузку. Математическая запись POLY2 имеет вид:

$$\hat{y}(x) = \omega_0 + \sum_{i=1}^n \omega_i x_i + \sum_{i=1}^n \sum_{j=i+1}^n w_{ij} x_i x_j,$$

где n – число признаков, ω_0 – глобальное смещение, w_i – вес i -го признака, w_{ij} – вес пересечения признаков i и j , x_i и x_j – соответствующие значения признаков i и j .

В дальнейшем Рендл с соавт. [8] предложили модель факторизационных машин (Factorization Machine, FM). FM эффективно выявляет сложные взаимодействия между признаками и повышает точность рекомендаций. За счет искусного сочетания идей матричной факторизации и пересечений второго порядка¹ FM успешно решает проблемы традиционных моделей машинного

¹ Факторизационные машины автоматически моделируют все возможные парные взаимодействия между любыми признаками (даже если их тысячи) без необходимости вручную создавать признаки вида «признак i × признак j ». – Прим. ред.

обучения при работе с большими разреженными наборами данных. По сути, FM представляет собой линейную модель первого порядка плюс все парные комбинации признаков с факторизацией весов:

$$\hat{y}(x) = \omega_0 + \sum_{i=1}^n \omega_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n \langle v_i, v_j \rangle x_i x_j,$$

где в $\sum_{i=1}^{n-1} \sum_{j=i+1}^n \langle v_i, v_j \rangle x_i x_j$ представлена часть перекрестной комбинации признаков, v_i – латентный вектор i -го признака, $\langle v_i, v_j \rangle$ – скалярное произведение латентных векторов v_i и v_j размерности k , $\langle v_i, v_j \rangle = \left\langle v_i, v_j \right\rangle = \sum_{l=1}^k v_{i,l} \cdot v_{j,l}$.

Вводя латентные векторы признаков, FM сокращает размер весовых параметров до $n \times k$ (где k – размер латентного вектора, а $n \gg k$), значительно снижая вычислительные затраты модели и затраты на обучение.

На базе FM Хуан и др. [9] предложили факторизационную машину с учетом полей (Field-aware Factorization Machine, FFM), в которой каждый признак обладает отдельным латентным вектором для каждого поля (field). Это еще сильнее повышает способность моделировать пересечения признаков. Формула FFM следующая:

$$\hat{y}(x) = w_0 + \sum_{i=1}^n w_i x_i + \sum_{i=1}^n \sum_{j=i+1}^n \langle v_{i,f_j}, v_{j,f_i} \rangle x_i x_j,$$

где v_{i,f_j} – латентный вектор признака i в поле признака j , $\langle v_{i,f_j}, v_{j,f_i} \rangle$ – скалярное произведение латентных векторов, x_i и x_j – значения i -го и j -го признаков во входном векторе признаков, f_i и f_j – поля признаков¹ (feature fields), к которым относятся i -й и j -й признаки соответственно.

На рис. 2.8 показаны диаграммы пересечения признаков POLY2, FM и FFM. В табл. 2.1 сравниваются методы пересечения признаков, количество параметров и временная сложность трех моделей.

¹ Поля признаков в рекомендательных системах и моделях машинного обучения – это логические группы или категории признаков, объединяющие похожие или связанные характеристики данных (например, «пол пользователя», «возраст», «категория товара», «бренд», «время суток»). Каждое поле содержит множество возможных значений (категориальных или числовых), которые преобразуются в эмбединги или one-hot векторы. В моделях вроде FFM или DeepFM поля признаков позволяют учитывать не только взаимодействия внутри поля, но и пересечения между разными полями (field-aware interactions), что значительно повышает выразительную мощность модели по сравнению с обычной матричной факторизацией, эффективно моделируя сложные нелинейные зависимости в разреженных данных. – Прим. ред.

Таблица 2.1. Сравнение POLY2, FM и FFM [10]

Модель	Пересечение функций	Количество параметров	Временная сложность
POLY2	$\sum_{i=1}^n \sum_{j=i+1}^n w_{ij} x_i x_j$	$1 + n + n, n$ – размерность признаков	$O(n^2)$
FM	$\sum_{i=1}^{n-1} \sum_{j=i+1}^n \langle v_i, v_j \rangle x_i x_j$	$1 + n + nk, k$ – размерность латентного вектора	$O(nk)$
FFM	$\sum_{i=1}^n \sum_{j=i+1}^n \langle v_{i,f_j}, v_{j,f_i} \rangle x_i x_j$	$1 + n + n(F - 1)k, F$ – количество полей	$O(nkF)$

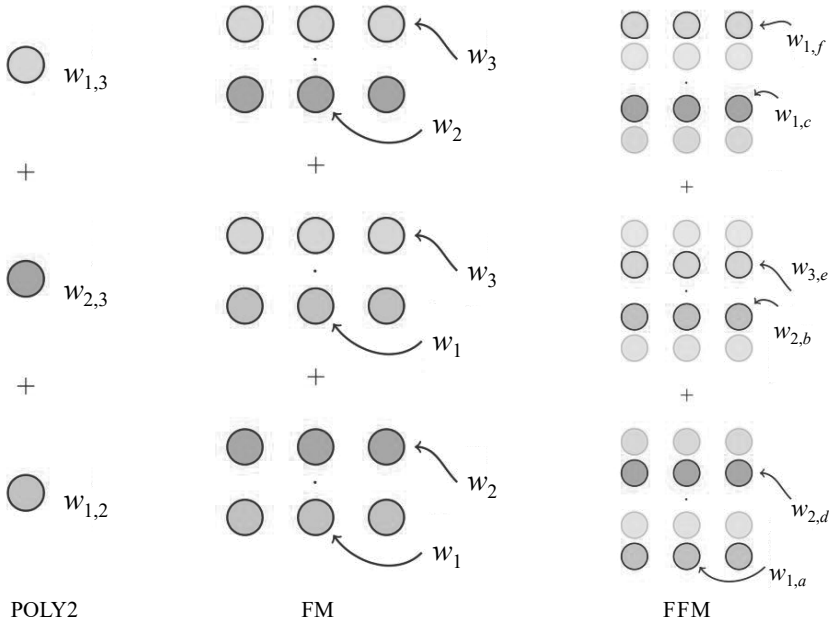


Рис. 2.8. Схема пересечения признаков в моделях POLY2, FM и FFM

Модельные ансамбли: интегрированное обучение

При построении рекомендательных систем разные алгоритмы обладают собственными сильными сторонами. Комбинируя их, можно взаимно компенсировать недостатки и существенно повысить эффективность и применимость системы. Ансамблирование моделей стало одним из ключевых практических достижений индустрии рекомендательных систем за последние годы. Оно позволяет выполнять сквозное обучение¹ (end-to-end learning), минимизиро-

¹ Сквозное обучение в рекомендательных системах – это подход, при котором вся цепочка обработки данных – от сырых входных сигналов (история взаимодействий, признаки пользователей и элементов, текст, изображения) до финального предсказания рекомендаций или ранжированного списка – обучается совместно в единой нейронной модели, без отдельных ручных этапов отбора кандидатов, конструирования признаков или промежуточного ранжирования. – *Прим. ред.*

вать ручной труд и создавать единый глобальный модельный фреймворк¹ для различных доменов (предметных областей) и бизнес-сценариев.

(1) Смешанная логистическая регрессия

Модель смешанной логистической регрессии² (Mixed Logistic Regression, MLR, также известная как LS-PLM) [11] была разработана в Alibaba и получила широкое распространение в индустрии. Сравнение качества подгонки³ MLR и обычной LR представлено на рис. 2.9. Архитектура MLR предельно прозрачна и реализует принцип «разделяй и властвуй»: сначала выполняется кластеризация без учителя образцов, затем в каждом кластере независимо обучается собственная логистическая регрессия. Такой подход обеспечивает значительно более точное предсказание CTR для разных групп пользователей и сценариев использования (рис. 2.10).

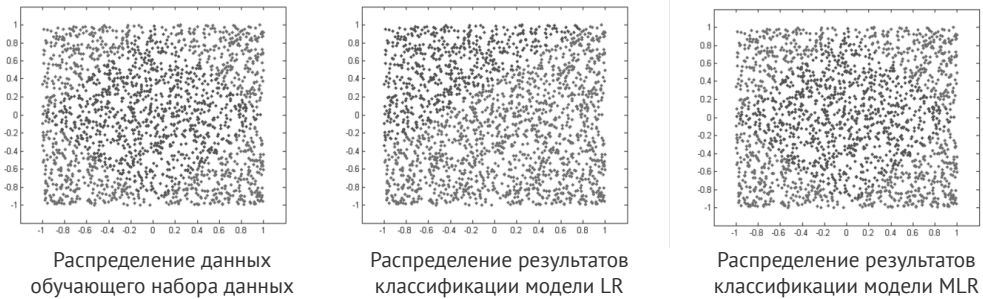


Рис. 2.9. Сравнение качества подгонки моделей MLR и LR

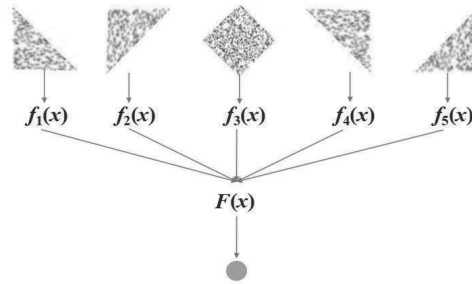


Рис. 2.10. Схема структуры модели MLR

¹ Фреймворк (framework) – это каркас, основа или инфраструктура для разработки программного обеспечения. Это готовый набор библиотек, инструментов, соглашений и архитектурных решений, который определяет структуру приложения и упрощает создание сложных систем, позволяя разработчику сосредоточиться на уникальной бизнес-логике, а не на решении рутинных, типовых задач. Примеры в веб-разработке: Django, Ruby on Rails, Spring, в машинном обучении: PyTorch, TensorFlow, Scikit-learn. – Прим. ред.

² Иногда используются варианты перевода «многоуровневая логистическая регрессия» или «иерархическая логистическая регрессия», это подчеркивает ключевую особенность модели: учет иерархической (многоуровневой) структуры данных. – Прим. перев.

³ Качество аппроксимации модели к данным, то есть насколько хорошо модель описывает и воспроизводит наблюдаемые данные (как точно предсказанные значения соответствуют реальным). – Прим. ред.

MLR обладает мощной нелинейной выразительной способностью¹ и встроенным отбором признаков, а также умеет извлекать обобщаемые нелинейные паттерны из больших разреженных наборов данных. Основная цель разработки – преодолеть ограничения масштабных обобщенных линейных моделей: недостаточную производительность, чрезмерную зависимость от ручного конструирования признаков и сложность кросс-доменного применения. До широкого распространения глубокого обучения MLR являлась одним из наиболее передовых подходов к моделированию рекомендательных систем.

(2) GBDT+LR

Модель GBDT+LR [12] представляет собой комбинацию градиентного бустинга² над деревьями решений³ (GBDT⁴) и логистической регрессии. Была предложена Facebook (ныне Meta) в 2014 году и демонстрирует превосходящее качество предсказания CTR по сравнению с использованием GBDT или LR по отдельности. Структура модели показана на рис. 2.11. На первом этапе множество деревьев GBDT выполняет отбор признаков и формирование высокоуровневых комбинаций из исходного вектора, порождая новый дискретный вектор признаков. Этот вектор затем подается на вход легкой модели LR, которая осуществляет финальное предсказание вероятности клика.

¹ Способность модели описывать и аппроксимировать сложные, в том числе нелинейные, зависимости в данных. Выразительная способность – это мера того, насколько богатым является класс функций, которые может представить (выразить) данная модель. – *Прим. ред.*

² Градиентный бустинг (gradient boosting) – это мощный метод машинного обучения для задач регрессии и классификации, основанный на последовательном построении ансамбля слабых моделей (обычно деревьев решений). Каждая следующая модель обучается исправлять ошибки предыдущих, минимизируя функцию потерь с помощью градиентного спуска в пространстве функций. Классические реализации: XGBoost, LightGBM, CatBoost. Это один из самых эффективных алгоритмов на структурированных данных. – *Прим. ред.*

³ Деревья решений (decision trees) – это модель машинного обучения для задач классификации и регрессии, представляющая решения в виде древовидной структуры. Модель начинается с корневого узла (весь датасет) и последовательно разбивает данные по признакам на ветви, выбирая на каждом шаге признак и порог, которые лучше всего разделяют классы (по критериям вроде Gini, энтропии или MSE). Листья дерева содержат финальные предсказания (класс или значение). Это интуитивно понятный, интерпретируемый метод, не требующий нормализации данных, но склонный к переобучению (решается ансамблями вроде случайного леса или градиентного бустинга). – *Прим. ред.*

⁴ GBDT (Gradient Boosting Decision Trees) – это метод машинного обучения, который используется для решения задач классификации и регрессии. Он основывается на алгоритме бустинга, который комбинирует предсказания нескольких деревьев решений для получения более точного результата. В отличие от логистической регрессии, которая является статистической моделью, использующей линейную комбинацию признаков для предсказания вероятности принадлежности к определенному классу, GBDT работает на основе ансамблевых методов и строит последовательность деревьев решений, каждое из которых обучается на остатках от предыдущих деревьев. Этот подход позволяет GBDT эффективно моделировать сложные нелинейные зависимости между признаками и целевой переменной. – *Прим. ред.*

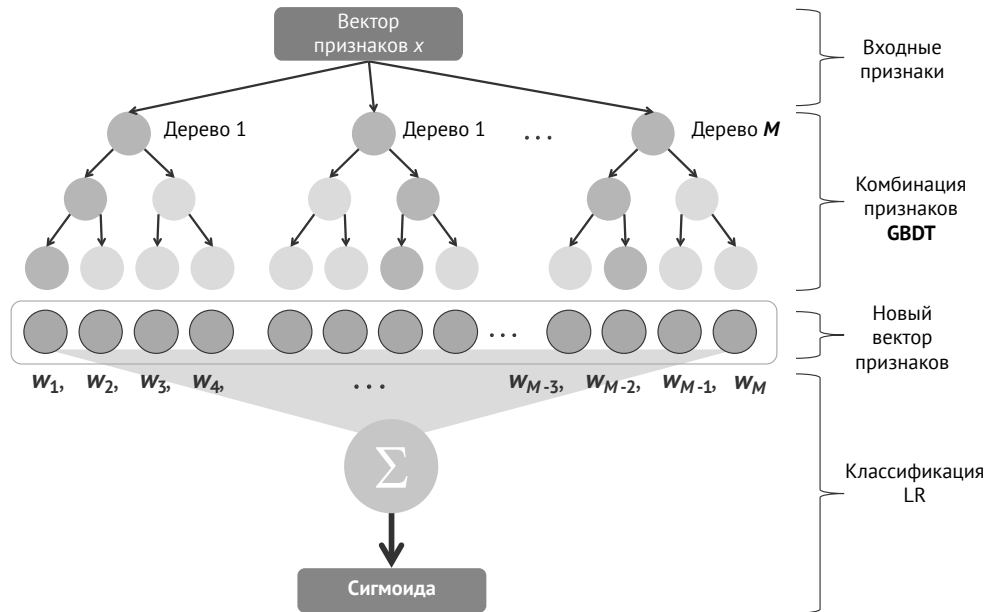


Рис. 2.11. Структура модели GBDT+LR

2.3.2. Методы рекомендаций на основе глубокого обучения

Развитие технологий глубокого обучения в области рекомендательных систем идет стремительными темпами. С 2015 года алгоритмы рекомендаций на базе глубоких сетей непрерывно эволюционируют, основное внимание уделяется эффективному вычислению представлений пользователей и элементов, а также оптимальным способам их комбинирования. Это направление стало главным вектором применения глубокого обучения в промышленных рекомендательных системах [13].

DSSM

Глубокая структурированная семантическая модель (Deep Structured Semantic Model, DSSM) [14], предложенная Microsoft и известная как классическая двухбашенная архитектура (Two-Tower Architecture), стала одним из первых успешных примеров глубокого обучения в рекомендательных системах. Изначально DSSM разрабатывалась для вычисления семантического сходства текстов через эмбединг-кодирование и косинусное расстояние, а технология хеширования слов¹ (word hashing) позволила обойти проблему огромного и разреженного словаря.

¹ Техника представления слов для обработки естественного языка, при которой каждое слово разбивается на набор перекрывающихся символовых n -грамм (чаще всего триграмм, например #ко, кот, от# для слова «кот»), а затем эти n -граммы преобразуются в вектор (bag-of- n -grams) с помощью хеш-функции в пространство фиксированной размерности. Это позволяет эффективно обрабатывать слова, не входящие в словарь, так как их n -граммы, как правило, уже встречались в обучении, и резко сокращает размерность матрицы эмбедингов по сравнению с «one-hot» кодированием по полному словарю.

В задаче полноты отбора рекомендательных систем двухбашенная структура DSSM (Dual-Tower) превращается в независимые башню пользователя (user tower) и башню элемента (item tower) (рис. 2.12). При онлайн-выводе (инференсе) достаточно извлечь эмбединги из обеих башен, вычислить их сходство и вернуть топ- n элементов. Модель поддерживает два режима отбора: $u2i$ (user-to-item) и $i2i$ (item-to-item).

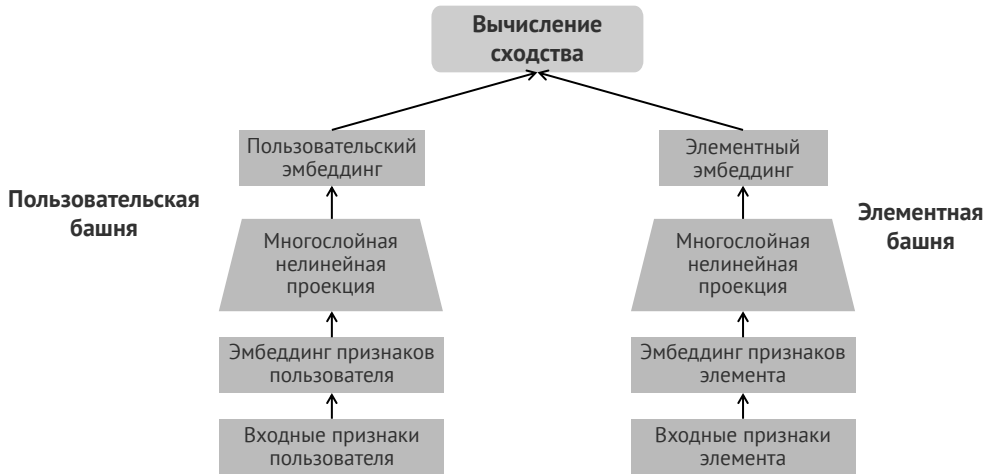


Рис. 2.12. Двухбашенная структура DSSM в сценарии отбора кандидатов

Из-за использования подхода «мешок слов»¹ (BoW, Bag-of-Words) модель DSSM теряет информацию о порядке слов и контексте. Кроме того, слабый контроль обучения, сквозной (end-to-end) режим и softmax на огромном количестве классов приводят к сильному смещению выборки в пользу популярных элементов. В 2019 году Google представила улучшенную двухбашенную модель [15], устранившую проблему смещения softmax за счет внутрипакетного softmax (in-batch softmax) и оценки частот потоковых данных. Модель успешно применена в рекомендательной системе видео YouTube (рис. 2.13).

¹ Подход «мешок слов» – это простая и широко используемая модель представления текста в обработке естественного языка и машинном обучении, где текст (предложение, документ) преобразуется в вектор фиксированной длины, игнорируя грамматику, порядок слов и синтаксис. Каждое уникальное слово из словаря корпуса становится отдельным измерением вектора, а значение в этом измерении отражает частоту появления слова в тексте (обычно TF – Term Frequency) или взвешенную частоту (TF-IDF). Таким образом, текст представляется как «мешок» (множество) слов без учета их последовательности, что позволяет эффективно применять классические алгоритмы ML (логистическая регрессия, SVM, Naïve Bayes) для задач классификации, кластеризации и поиска похожих документов, хотя модель теряет информацию о контексте и семантике. – Прим. ред.

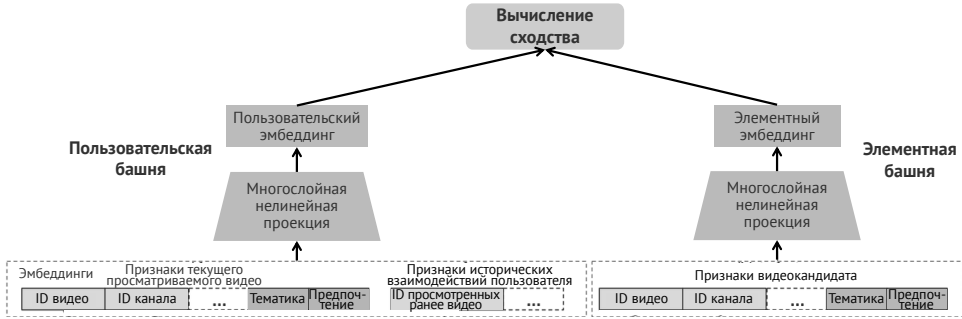


Рис. 2.13. Улучшенная двухбашенная модель в рекомендательной системе видео YouTube

Wide & Deep

Модель Wide & Deep, предложенная Google [16], оказала огромное влияние на индустрию благодаря изящному балансу между запоминанием (memorization) и обобщением (generalization).

Структура модели показана на рис. 2.14. Широкая часть – это обобщенная линейная модель (по сути, логистическая регрессия), которая эффективно запоминает редкие низкоуровневые комбинации признаков при минимальном числе параметров. Глубокая часть – классическая полносвязная нейронная сеть (DNN), которая через слои векторных представлений (эмбединг-слои) преобразует категориальные признаки в плотные векторы и извлекает сложные высокоуровневые закономерности.

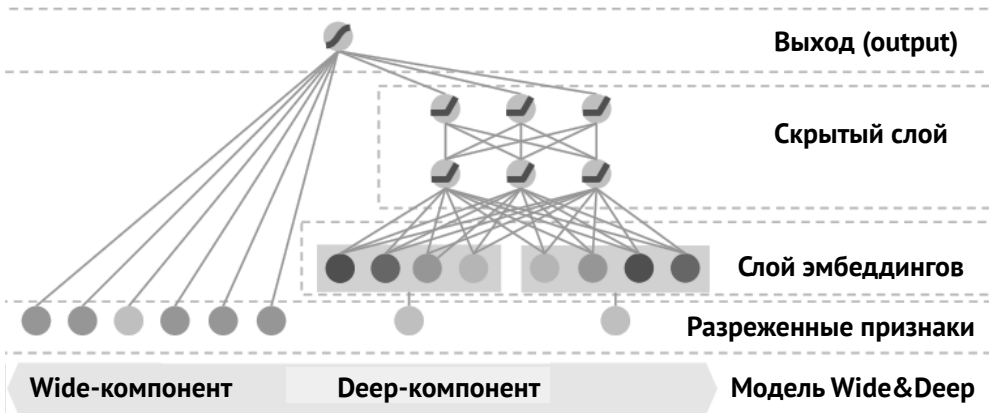


Рис. 2.14. Структура модели Wide&Deep

Модель Wide&Deep использует совместное обучение (joint training): в процессе обучения параметры широкой и глубокой компонент обновляются одновременно. Как было сказано выше, Wide&Deep объединяет сильные стороны DNN и LR и может быть представлена в виде:

$$P(Y = 1 | \mathbf{x}) = \sigma \left(\mathbf{w}_{\text{wide}}^T [\mathbf{x}, \phi(\mathbf{x})] + \mathbf{w}_{\text{deep}}^T a^{(l_r)} + b \right).$$

Модель Wide&Deep умело сочетает в себе способность запоминать историческое поведение с ее способностью к обобщению и анализу новых ситуаций. При этом ее парадигма построения фреймворка продолжает оказывать значительное влияние на алгоритмы рекомендаций глубокого обучения на основе признаков. Многие последующие алгоритмы рекомендаций представляют собой усовершенствования фреймворка Wide&Deep.

Модель DeepFM (Deep Factorization Machine) [17], разработанная Huawei и Харбинским технологическим институтом в 2017 году, представляет собой важное развитие Wide&Deep, частично за счет замены Wide-компонента на FM-модуль (рис. 2.15). Deep-часть остается без изменений и отвечает за высокоуровневые взаимодействия. Обе ветви используют общий входной слой и слой эмбедингов.

Модель DeepFM объединяет преимущества DNN и FM, позволяя одновременно обучать низкоуровневые и высокоуровневые взаимодействия признаков и преодолевая ограничение классической FM, которая учитывает только комбинации второго порядка. Благодаря такой архитектуре DeepFM реализует полностью сквозное конструирование признаков без какого-либо ручного вмешательства. Результат предсказания модели DeepFM можно записать в виде:

$$\hat{y} = \text{sigmoid}(y_{\text{FM}} + y_{\text{DNN}});$$

$$y_{\text{FM}} = w_0 + \sum_{i=1}^n w_i x_i + \sum_i \sum_{j=i+1}^n \langle v_i, v_j \rangle x_i x_j;$$

$$y_{\text{DNN}} = \sigma(W^{|H|+1} \cdot a^H + b^{|H|+1}).$$

DeepFM не только снижает нагрузку по конструированию признаков, но и превосходит модель Wide&Deep на некоторых наборах данных.

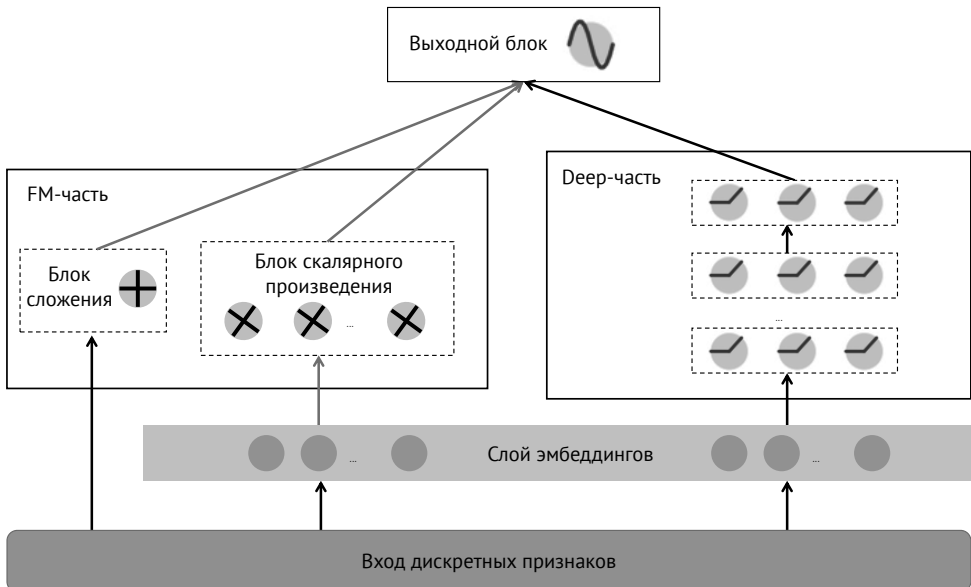


Рис. 2.15. Структура модели DeepFM

данные в информативные, компактные и интерпретируемые переменные, которые модель сможет эффективно использовать.

Качественные признаки ускоряют сходимость, повышают точность и устойчивость модели. Основные инструменты: sklearn, Pandas, NumPy, Featuretools и др. Правильная обработка данных – фундаментальное условие точной и эффективной работы рекомендательной системы.

Типы признаков

Признаки можно разделить на такие типы, как базовые, статистические и последовательностные. Базовые признаки, такие как основная информация о пользователе или описательные данные объекта, имеют широкую область применения и могут использоваться в различных бизнес-сценариях. Статистические признаки в основном являются апостериорными и представляют собой количественное выражение поведения пользователей или характеристик элементов, помогая модели понять интересы пользователей и степень популярности элементов. Последовательностные признаки, в свою очередь, используются для представления временных отношений во взаимодействиях «пользователь–элемент». Далее признаки будут рассмотрены по трем основным группам: пользовательские, элементные и контекстные.

(1) Признаки пользователя (портрет пользователя)

Пользовательские признаки, или портрет пользователя, включают несколько основных групп.

- Базовые признаки. Демографические данные (ID, пол, возраст, образование, город проживания, профессия) – собираются сложно и редко меняются; системные данные (время регистрации, статус VIP, признак нового пользователя, уровень аккаунта и т. д.). Частота изменения низкая.
- Статистические признаки. Агрегации, полученные на основе логов поведения пользователя (клики, покупки, добавления в корзину и т. д.). Важно учитывать одновременно абсолютные величины и пропорции (например, не только «количество покупок», но и «доля покупок в категории», «CTR пользователя за последние 7 дней» и т. п.).
- Последовательностные признаки поведения. Хронологические последовательности действий пользователя: тип действия (клик, покупка, комментарий, просмотр и т. д.) + связанные элементы (товар, категория, бренд). Делятся на краткосрочные (последние часы/дни) и долгосрочные (недели/месяцы) последовательности.
- Признаки социальных связей пользователя. Информация о социальных связях или степени схожести с другими пользователями (граф дружбы, похожие группы, кластеры похожих пользователей).

(2) Признаки элемента

Элемент – ключевая сущность рекомендательного акта. Глубокая проработка его признаков напрямую повышает точность и уровень персонализации. Признаки элемента делятся на следующие основные типы.

- Базовые признаки. Зависят от домена. В e-commerce: название, описание, бренд, цена, материал, цвет и т. д. В видеорекомендациях: название,

описание, жанр, длительность, автор/канал, год выпуска и т. п. Заполняются вручную или генерируются автоматически (NLP, CV-модели).

- Статистические признаки. Отражают популярность, эффективность и качество элемента: количество показов, кликов, конверсий, положительных отзывов, CTR, конверсионная ставка, рейтинг и т. д. Дополнительно рассчитываются статистики по категориям, брендам, источникам контента.
- Перекрестные признаки. Комбинации двух и более базовых/статистических признаков (например, «категория × CTR за последние 24 ч», «бренд × средняя цена в регионе», «жанр × средняя длительность просмотра»). Позволяют модели захватывать более сложные и тонкие закономерности.

(3) Контекстные признаки

Контекстные признаки описывают среду, в которой происходит пользовательское действие. Их учет позволяет модели гораздо тоньше понимать текущие потребности и значительно повышает точность и пользовательский опыт. Основные группы контекстных признаков следующие.

- Сцена контекста. Точка входа и раздел приложения: главная страница, детальная страница товара, результаты поиска, операционная активность, пуш-уведомления и т. д. Для каждой сцены обычно требуется отдельная стратегия рекомендаций.
- Временной контекст. Праздники, сезонность, день недели, время суток. Один и тот же пользователь может кардинально менять предпочтения в рабочие часы в офисе и вечером дома (например, при заказе еды).
- Географический контекст. Текущая геопозиция и регион проживания. Климат, культурные особенности и локальные тренды сильно влияют на спрос (одежда, еда, развлечения).
- Контекст устройства. Тип устройства (смартфон, планшет, десктоп), тип сети (Wi-Fi / мобильный интернет), версия клиента (iOS/Android, конкретный билд приложения или веб-версия). Влияет как на форму подачи рекомендаций, так и на вероятность конверсии.

Методы обработки признаков

Обработка признаков включает в себя несколько этапов, таких как обработка пропущенных значений, преобразование признаков, отбор признаков и комбинирование признаков. Во-первых, обработка пропущенных значений в признаках является основой для обеспечения целостности данных. Затем преобразование признаков выполняет соответствующее изменение исходных признаков, чтобы адаптировать их к формату входных данных модели, сохраняя при этом внутренние закономерности данных и облегчая обучение модели. Отбор признаков, в свою очередь, путем сравнения и оценки позволяет выбрать из множества признаков те, которые наиболее полезны для эффективности модели, тем самым упрощая модель и повышая эффективность ее обучения и вывода. Наконец, путем комбинирования нескольких признаков создается информация, имеющая большую бизнес-ценность.

(1) Обработка пропущенных значений

Первый шаг в обработке признаков – работа с пропущенными значениями (missing values). Наиболее распространенные методы следующие.

- Прямое удаление. Удаляем признак, если пропусков > 50 % (и он не критичен). Удаляем пример, если пропуск в важном признаке и таких семплов < 0,1 %. Не подходит для маленьких наборов данных.
- Заполнение константой. Присваиваем 0, –1 или специальное значение (например, «unknown»). Просто, но может вносить шум.
- Заполнение статистикой. Среднее, медиана, мода. Самый популярный и быстрый подход, но может исказить распределение.
- Прямое/обратное заполнение (Forward/Backward fill). Берем предыдущее или следующее известное значение. Идеально для временных рядов.
- Интерполяция. Линейная, полиномиальная и др.
- Заполнение с помощью предсказания модели. LR, Random Forest, CatBoost и др. обучаются на известных значениях и предсказывают пропуски.

(2) Трансформация признаков

Преобразование признаков – это применение математических методов и функций к исходным признакам с целью получения более информативных представлений. Признаки делятся на три основных класса, каждый из которых требует своего подхода к обработке: категориальные (дискретные, *англ.* discrete/categorical features), непрерывные (continuous), многозначные (признаки с множественными значениями, *англ.* multi-value).

Категориальные признаки обычно требуют специального кодирования, чтобы стать пригодными для большинства моделей [34]. После такого кодирования (например, унитарного кодирования) они часто приобретают разреженное представление. Наиболее распространенные методы кодирования следующие.

- Бинарное кодирование (binary encoding): подходит для признаков с только двумя категориями, например является ли пользователь VIP (is_VIP); такие признаки можно представить как 0 или 1.
- Кодирование метками (label encoding): каждый нечисловой класс преобразуется в числовую метку (например, номер от 1 до N), но этот способ вводит искусственный порядок, что может негативно повлиять на работу модели. Например, метки пользователя «любит читать книги», «любит снеки» и «любит спорт» можно закодировать соответственно как 1, 2 и 3. Внимание: создает ложную упорядоченность, используем с осторожностью.
- Унитарное кодирование (one-hot encoding): каждый класс преобразуется в уникальный двоичный вектор; этот метод устраняет проблему искусственного порядка, создаваемую кодированием метками, но увеличивает размерность пространства признаков. Например, для разных цветов «красный» можно закодировать как [1, 0, 0, 0], а «желтый» – как [0, 1, 0, 0].
- Эмбединговые представления: дискретные признаки отображаются в низкоразмерные векторы; подход подходит для случаев с большим числом категорий или сложными взаимосвязями. К распространенным методам получения векторных представлений относятся word2vec и GloVe (на основе семантики текста), а также DeepWalk и node2vec (на основе графов знаний).

Непрерывные признаки (continuous features), также называемые *плотными*, обычно обрабатываются с помощью нормализации, стандартизации и дискретизации. Все эти приемы относятся к технологиям масштабирования признаков (feature scaling) [34] и позволяют привести признаки к виду, оптимальному для обучения моделей.

- Нормализация (Min-Max scaling). Масштабирует значения в фиксированный диапазон (обычно [0, 1] или [-1, 1]). Сохраняет исходную форму распределения и эффективно устраняет различия в размерностях признаков.
- Стандартизация (Z-score normalization). Приводит распределение к стандартному нормальному виду: среднее = 0, стандартное отклонение = 1. Особенно полезна для алгоритмов, чувствительных к масштабу (линейные модели, SVM, нейронные сети), и хорошо справляется с выбросами.
- Дискретизация (Binning). Разбиение непрерывного признака на интервалы (равной ширины, равной частоты или кластерный биннинг). Упрощает модель, помогает захватывать нелинейные зависимости и защищает от переобучения на аномальных значениях.

Многочленные признаки (multi-value features) состоят из переменного числа элементов (классические примеры – последовательности действий пользователя и списки тегов). Из-за разной длины их нельзя подать в модель напрямую, поэтому требуется преобразование к фиксированному или близкому размеру. Ниже приведены наиболее распространенные методы.

- Подсчет элементов (Element Counting / Frequency Encoding). Частота появления каждого элемента. Пример: распределение покупок по категориям можно представить как {"книги": 3, "спорт": 1, "снеки": 1}.
- Кодирование метками (Label Encoding). Каждому элементу присваивается уникальный числовой ID. Пример: ["клик", "добавление в корзину", "оплата", "комментарий"] можно закодировать как [1, 2, 3, 4].
- Мультихот-кодирование (multi-hot encoding), или многомерное «горячее» кодирование. Каждый элемент преобразуется в one-hot-вектор по общему словарю, при этом несколько единиц допустимо. Сохраняет порядок и множественность. Пример: теги фильма ["триллер", "боевик"] при словаре ["триллер", "комедия", "боевик", "мелодрама", "документальный", "фантастика"], после мультихот-кодирования вектор для этого фильма будет [1, 0, 1, 0, 0, 0].
- Векторное представление текста (text vector representation). Использование методов CountVectorizer, TfidfVectorizer или встраиваний (embeddings) для преобразования текстовых данных в векторы и извлечения их семантической информации. Пример: заголовок товара «Юньнаньский фильтр-кофе эспрессо молотый» → {"Юньнань": 1, "фильтр": 1, "кофе": 2, "эспрессо": 1, "молотый": 1}.
- Эмбединг-представления. Каждый элемент (или вся последовательность) отображается в плотный низкоразмерный вектор. Пример: последовательность действий [клик, покупка, добавление в корзину] можно преобразовать в вектор частот слов: [[0.2, 0.5, 0.8], [0.9, 0.3, 0.6], [0.4, 0.7, 0.1]] (обычно получают через item2vec, BERT-подобные модели, RNN/Трансформер-кодировщики и т. п.).

(3) Отбор признаков

Отбор признаков (feature selection) – это выбор из полного набора только тех признаков, которые релевантны целевой задаче. Он позволяет избавиться от неинформативных и избыточных переменных, упростить модель, ускорить обучение и инференс, а главное – повысить обобщающую способность модели. В процессе отбора необходимо оценивать важность, коррелированность и влияние каждого признака на стабильность модели [36].

Методы отбора признаков традиционно делятся на три большие группы.

- Методы фильтрации (filter methods). Оценивают признаки независимо от модели с помощью статистических или информационно-теоретических критериев: χ^2 -тест, коэффициенты корреляции, дисперсионный анализ, информационный выигрыш (information gain), взаимная информация (mutual information) и др. Преимущества: крайне низкая вычислительная стоимость. Недостаток: выбранный набор не всегда совпадает с оптимальным для конкретной модели.
- Методы-обертки (wrapper methods). Выбирают различные подмножества признаков, обучают модель на основе целевой функции, проверяют ее прогнозную эффективность (predictive performance), тем самым оценивая, в какой степени каждое подмножество признаков улучшает производительность модели. Методы-обертки обычно используют такие стратегии поиска, как жадный поиск (greedy search), генетические алгоритмы, рекурсивное исключение признаков (Recursive Feature Elimination, RFE), для нахождения оптимального подмножества признаков. Хотя методы-обертки обладают высокой точностью, их вычислительная сложность значительно выше, чем у методов-фильтров, поскольку требуют повторения шагов обучения и перекрестной проверки (cross-validation).
- Эмбединговые методы (embedded methods). Отбор происходит непосредственно в процессе обучения модели за счет встроенной регуляризации или механизма важности признаков. Классические примеры: L1- и L2-регуляризация в линейных моделях, встроенная важность признаков в деревьях решений (GBDT, Random Forest, XGBoost, LightGBM и др.). Преимущества: умеренные вычислительные затраты и тесная интеграция с моделью. Недостаток: привязанность к конкретному алгоритму.

В рекомендательных системах с сотнями тысяч и миллионами признаков особенно критично проверять отобранный набор на валидационной выборке, чтобы гарантировать реальное повышение качества модели, а не просто сокращение размерности.

(4) Комбинирование признаков

Комбинирование признаков – один из самых эффективных способов повысить производительность модели. Путем объединения двух или более признаков определенным способом можно выявить сложные взаимосвязи, которые могут существовать между исходными признаками. Ниже представлены несколько распространенных методов комбинации признаков.

- Линейные комбинации (linear combination). Самый простой вариант – создание нового признака как взвешенной суммы существующих через линейную функцию.
- Полиномиальные комбинации (polynomial combination). Это метод нелинейной комбинации признаков, который позволяет уловить более сложные эффекты взаимодействия между признаками за счет введения полиномиальных признаков.
- Перекрестная комбинация признаков (feature crossing / feature cross). Это распространенный метод обработки категориальных признаков, при котором два или более признака органично комбинируются для создания нового составного признака. Цель перекрестной комбинации – уловить взаимодействия между категориальными признаками, построить более сильные синтетические признаки, снизить сложность обучения модели и позволить ей сосредоточиться на изучении другой важной информации.

Реализация конструирования признаков

С ростом мощности моделей подход к конструированию признаков постепенно смещается от полностью ручного к автоматизированному. Оба подхода имеют свои сильные стороны и области применения.

Ручное конструирование признаков остается стандартом при небольших наборах данных и относительно простых моделях. Специалист, глубоко понимающий бизнес, проводит EDA¹, выявляет ключевые закономерности и вручную конструирует высокоинформативные признаки. Преимущества: максимальная точность признаков, простота модели, высокая скорость инференса, прозрачность и легкость поддержки.

Автоматизированное конструирование признаков получает все большее распространение по мере роста объемов данных, сложности моделей и доступной вычислительной мощности. Алгоритмы автоматически генерируют, отбирают и комбинируют тысячи кандидатов (Featuretools, глубокие архитектуры с end-to-end обучением, AutoML-системы). Особенно ярко преимущество проявляется в автоматическом создании перекрестных и полиномиальных признаков через нейронные сети (FM, DeepFM, AutoInt и др.). Плюсы: обработка гигантских наборов данных, снижение человеческого фактора, возможность сквозной оптимизации; минусы: рост сложности и вычислительных затрат.

Несмотря на успехи автоматизации, она пока не способна полностью заменить экспертное знание домена. Лучшие производственные системы практи-

¹ EDA (Exploratory Data Analysis) – это разведочный анализ данных, этап предварительного исследования набора данных с целью выявления основных закономерностей, аномалий, распределений, корреляций и скрытых структур. В контексте машинного обучения и рекомендательных систем специалист проводит EDA с помощью статистических сводок, визуализаций (гистограммы, боксплоты, корреляционные матрицы, scatter-плоты и т. д.) и простых расчетов, чтобы глубже понять данные, обнаружить важные зависимости, очистить выбросы и сформировать гипотезы для последующего конструирования признаков или выбора модели. EDA позволяет избежать слепого применения алгоритмов и создать более информативные и точные признаки вручную, особенно когда набор данных небольшой и бизнес-домен хорошо понятен. – Прим. ред.

чески всегда используют гибридный подход: автоматические пайплайны как основа + тонкая ручная доработка критически важных признаков. Поэтому глубокое понимание принципов ручного конструирования признаков остается обязательным навыком для любого специалиста по рекомендательным системам.

2.6. ЗАКЛЮЧЕНИЕ

Рекомендательные системы – это сложная, многогранная и постоянно развивающаяся область, в которой переплетаются передовые алгоритмы, масштабная инженерия и глубокое понимание пользовательского поведения. Настоящая глава дала целостное представление о проектировании, реализации и оценке современных персонализированных рекомендательных систем: от классических алгоритмов до промышленных архитектурных решений. Изложенный материал создает прочную базу для дальнейшего изучения применения больших моделей в рекомендательных системах – тема, которой будут посвящены последующие главы.

Для новичков эта глава может служить ускоренным введением в мир рекомендательных систем. Для читателей с опытом – систематизацией и актуализацией знаний. Только уверенное владение фундаментальными принципами и инженерными практиками позволяет в полной мере понять и эффективно использовать потенциал больших моделей в задачах рекомендаций.

СПИСОК ЛИТЕРАТУРЫ

- [1] Koren Y., Bell R.M., Volinsky C. Matrix Factorization Techniques for Recommender Systems [J]. Computer, 2009, 42.
- [2] Salakhutdinov R., Mnih A., Hinton G.E. Restricted Boltzmann machines for collaborative filtering [C], 2007.
- [3] Salakhutdinov R., Hinton G.E. Deep Boltzmann Machines [C], 2009.
- [4] Li S., Kawale J., Fu Y.R. Deep Collaborative Filtering via Marginalized Denoising Auto-encoder [J]. Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, 2015.
- [5] Xue H., Dai X., Zhang J., et al. Deep Matrix Factorization Models for Recommender Systems [C], 2017.
- [6] Richardson M., Dominowska E., Ragnó R. Predicting Clicks: Estimating the Click-Through Rate for New Ads: WWW2007; International World Wide Web Conference [C], 2007.
- [7] Chang Y., Hsieh C., Chang K., et al. Training and Testing Low-degree Polynomial Data Mappings via Linear SVM [J]. J. Mach. Learn. Res., 2010, 11: 1471–1490.
- [8] Rendle S. Factorization Machines [J]. IEEE, 2010.
- [9] Juan Y., Zhuang Y., Chin W.S., et al. Field-aware Factorization Machines for CTR Prediction: Conference on Recommender Systems [C], 2016.
- [10] Wang S., Yang M., Liu Z., et al. Comparison of Underlying Algorithms in Recommendation Systems [C], 2022.
- [11] Gai K., Zhu X., Li H., et al. Learning Piece-wise Linear Models from Large Scale Data for Ad Click Prediction [J]. 2017.

- [12] He X., Pan J., Jin O., et al. Practical Lessons from Predicting Clicks on Ads at Facebook [M]. Practical Lessons from Predicting Clicks on Ads at Facebook, 2014.
- [13] HUANG B., YAN F., ZHANG H., et al. Progress and Application of Recommendation System [J]. J Wuhan Univ (Nat Sci Ed), 2021, 67(6): 503–516.
- [14] Huang P.S, He X., Gao J., et al. Learning deep structured semantic models for web search using clickthrough data: Conference on Information and Knowledge Management [C], 2013.
- [15] Yi X., Yang J., Hong L., et al. Sampling-bias-corrected neural modeling for large corpus item recommendations [J]. Proceedings of the 13th ACM Conference on Recommender Systems, 2019.
- [16] Cheng H.T, Koc L., Harmsen J., et al. Wide & Deep Learning for Recommender Systems [J]. ACM, 2016.
- [17] Guo H., Tang R., Ye Y., et al. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction [J]. ArXiv, 2017, abs/1703.04247.
- [18] Song W., Shi C., Xiao Z., et al. AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks [J]. Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2018.
- [19] Rendle S., Freudenthaler C., Schmidt-Thieme L. Factorizing personalized Markov chains for next-basket recommendation [C], 2010.
- [20] Wang P., Guo J., Lan Y., et al. Learning Hierarchical Representation Model for NextBasket Recommendation: SIGIR '15 [C], New York, NY, USA, 2015.
- [21] Schmidt R.M. Recurrent Neural Networks (RNNs): A gentle Introduction and Overview.
- [22] Hidasi B.A.Z., Karatzoglou A., Baltrunas L., et al. Session-based Recommendations with Recurrent Neural Networks [J]. CoRR, 2015, abs/1511.06939.
- [23] Kang W.C., Mcauley J. Self-Attentive Sequential Recommendation: 2018 IEEE International Conference on Data Mining (ICDM) [C], 2018.
- [24] Zhou G., Song C., Zhu X., et al. Deep Interest Network for Click-Through Rate Prediction [J]. 2017.
- [25] Zhou G., Mou N., Fan Y., et al. Deep Interest Evolution Network for Click-Through Rate Prediction: National Conference on Artificial Intelligence [C], 2019.
- [26] Feng Y., Lv F., Shen W., et al. Deep Session Interest Network for Click-Through Rate Prediction [J]. 2019.
- [27] Pi Q., Bian W., Zhou G., et al. Practice on Long Sequential User Behavior Modeling for Click-Through Rate Prediction [J]. ACM, 2019.
- [28] Cheng Y., Wang D., Zhou P., et al. A Survey of Model Compression and Acceleration for Deep Neural Networks [J]. ArXiv, 2017, abs/1710.09282.
- [29] Graepel T., Candela J.Q.N.O., Borchert T., et al. Web-Scale Bayesian Click-Through rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine [C], 2010.
- [30] Opper M. A Bayesian Approach to Online-learning [C], 2006.
- [31] McMahan H.B., Holt G., Sculley D., et al. Ad click prediction: a view from the trenches [J]. Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, 2013.
- [32] Auer P., Cesa-Bianchi N.O., Fischer P. Finite-time Analysis of the Multiarmed Bandit Problem [J]. Machine Learning, 2002, 47: 235–256.

- [33] Zheng G., Zhang F., Zheng Z., et al. DRN: A Deep Reinforcement Learning Framework for News Recommendation [J]. Proceedings of the 2018 World Wide Web Conference, 2018.
- [34] Zheng A., Casari A. Feature Engineering for Machine Learning Models: Principles and Techniques for Data Scientists [M]. O'Reilly, 2019.
- [35] Zsoy M.G.U.L. From Word Embeddings to Item Recommendation [J]. ArXiv, 2016, abs/1601.01356.
- [36] Guyon I.M., Elisseeff A.E. An Introduction to Variable and Feature Selection [J]. J. Mach. Learn. Res., 2003, 3: 1157–1182.