2021

Лекция 13. Рекомендательные системы

МЕТОДЫ МАШИННОГО ОБУЧЕНИЯ

ПАПУЛИН С.Ю. (papulin.study@yandex.ru)

Дополнительные материалы доступны по ссылке: https://github.com/mlmethods

Содержание

1. Метод k-ближайших соседей

1.1. Классификация

Дано обучающее множество S такое, что

$$S = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$$

Необходимо построить классификатор, который каждому наблюдению ставит в соответствие некоторый класс из C.

При использовании для классификации метода k-ближайших соседей вероятность принадлежности некоторого нового наблюдения x_0 классу j вычисляется как

$$p(Y = j|X = x_0) = \frac{1}{K} \sum_{i \in N_0} I(y_i = j)$$

где N_0 – множество K индексов элементов обучающего набора, ближайших к x_0.

Предсказание класса для наблюдения x_0 осуществляется следующим образом:

$$\hat{y}_0 = \underset{y \in C}{\operatorname{argmax}}\, p(Y = y|X = x_0)$$

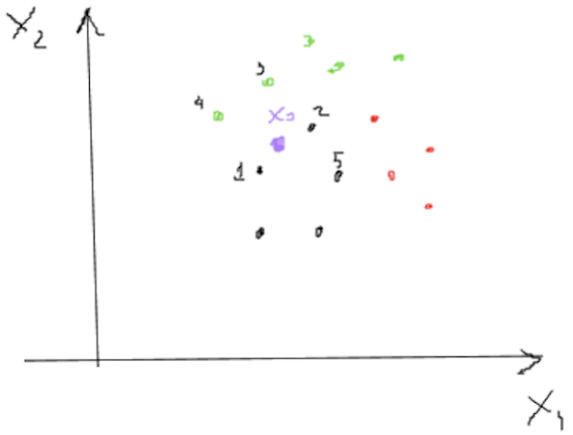


Рисунок – Пример k-ближайших соседей для задачи классификации: три класса $C = \{0,1,2\}$ и $K = 5$ соседей

1.2. Регрессия

В случае применения метода k-ближайших соседний к задаче регрессии значение предсказания для некоторого наблюдения x_0 будет вычисляться как

$$\hat{y}_0 = \hat{f}(x_0) = \frac{1}{K} \sum_{i \in N_0} y_i$$

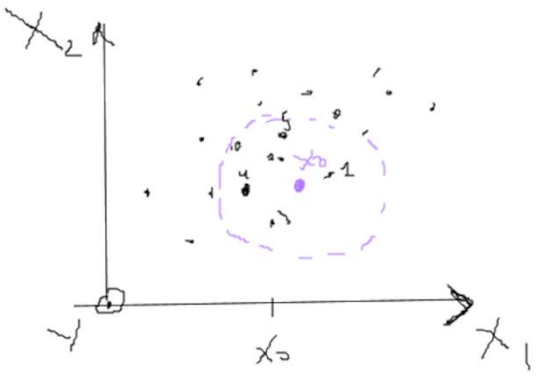


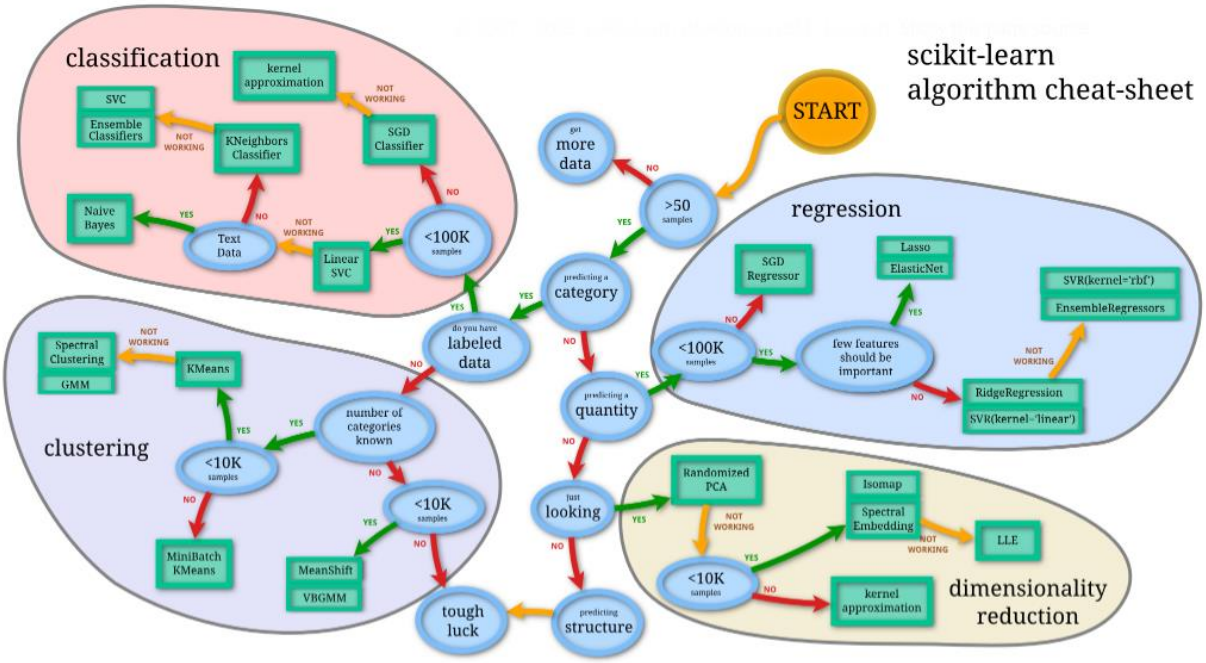
Рисунок – Пример k-ближайших соседей для задачи регрессии

Таким образом, в методе k-ближайших соседей отсутствует как таковая стадия обучения. По факту для каждого нового наблюдения x_0 метод должен выбрать K ближайших наблюдений из обучающего множества, по которым далее будет рассчитываться предсказание.

Наблюдения обучающего множества должны находиться в оперативной памяти для быстрого доступа. Кроме того, с ростом количества наблюдений увеличивается количество возможных вариантов соседей и, следовательно, время предсказания.

В отличие от ранее рассмотренных методов обучения, в которых обучения занимает значительную часть времени по сравнению с предсказанием, в методе k-ближайших соседей обратная ситуация.

Выбор подходящего метода



https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

2. Рекомендательные системы

2.1. Постановка задачи и основные обозначения

Можно выделить несколько основных подходов к построению рекомендательных систем:

- На основе контента
- Коллаборативная фильтрация
- Факторизация матрицы рейтингов (часто рассматривается как вид коллаборативной фильтрация)

Основные обозначения

$\mathcal{U} = \{u_1, u_2, \dots, u_m\}$	Пользователи
$\mathcal{I} = \{i_1, i_2, \dots, i_n\}$	Объекты/товары
$\mathcal{R}$	Матрица рейтингов размера $m \times n$
$R_{u,i}$	Элемент матрицы, обозначает рейтинг пользователя u на i-ый объект
U_i	Подмножество пользователей, которые проставили рейтинг некоторому товару i, $U_i \subseteq \mathcal{U}$
I_u	Подмножество товаров, которым пользователь u проставил рейтинги, $I_u \subseteq \mathcal{I}$
U_{ij}	Подмножество пользователей, которые проставили рейтинги обоим товарам i и j
I_{uv}	Товары, которым были проставлены рейтинги пользователями u и v, то есть $I_u \cap I_v$

Каждому пользователю u соответствует набор объектов I_u, о которых он выразил свое мнение (в явном виде – рейтинг, неявно – купил или нет, заходил на страницу товара или нет), $I_u \subseteq \mathcal{I}$

Две наиболее важные задачи рекомендательных системам являются определение лучшего товара и формирование первых N рекомендаций.

Основные задачи

- *Предсказание*. Определение степени заинтересованности пользователя u в объекте $i \notin I_u$, обозначается как $P_{u,i}$ или $\hat{R}_{u,i}$
- *Рекомендация*. Определение списка из N объектов, $I_r \subset \mathcal{I}$, которыми пользователь u будет наиболее заинтересован. При этом $I_r \cap I_u = \emptyset$

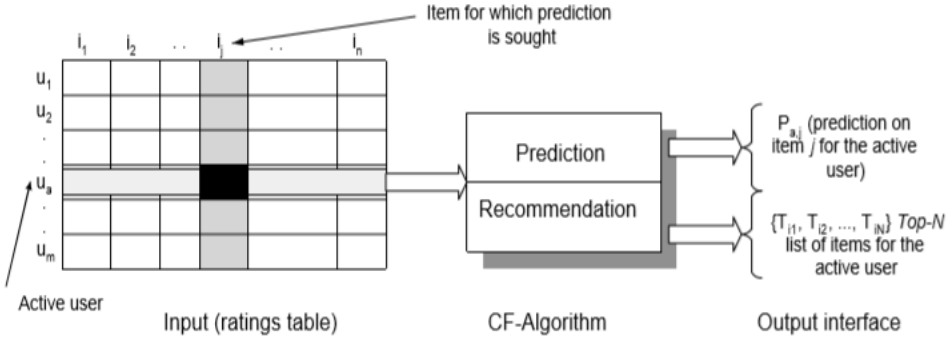


Рисунок – Задачи коллаборативной фильтрации [2]

Предсказание

Задача заключается в поиске для пользователя u нового товара $i \in \mathcal{I} \backslash I_u$, который с высокой вероятностью его заинтересует.

Когда доступны рейтинги, задачу предсказания можно определить как регрессию или многоклассовую классификацию. В этом случае необходимо обучить функцию $f: \mathcal{U} \times \mathcal{I} \to \mathcal{S}$ для предсказания рейтинга $f(u, i)$ пользователя u новому товару i.

Далее эта функция используется для *рекомендации* пользователю u_a товара i^*, для которого оценка рейтинга будет наивысшей:

$$i^* = \arg\max_{i \in \mathcal{I} \backslash I_u} f(u, i)$$

Как правило, матрица рейтингов $\mathcal{R}$ разбивается на обучающее множество $\mathcal{R}_{train}$ для обучения f и тестовое множество $\mathcal{R}_{test}$ для оценки качества предсказания с использованием средней абсолютной ошибки (MAE) или квадратного корня из среднеквадратической ошибки (RMSE):

$$\mathrm{MAE}(f) = \frac{1}{|\mathcal{R}_{test}|} \sum_{R_{ui} \in \mathcal{R}_{test}} |R_{ui} - f(u, i)|$$

и

$$\mathrm{RMSE}(f) = \sqrt{\frac{1}{|\mathcal{R}_{test}|} \sum_{R_{ui} \in \mathcal{R}_{test}} \big(R_{ui} - f(u, i)\big)^2}$$

3. Рекомендации на основе контента

3.1. Общие сведения

Большинства рекомендательных систем на основе контента используют относительно простые модели поисковых систем, например, по совпадению ключевых слов или модель векторного пространства (VSM) с TF-IDF преобразованием.

При использовании VSM текстовый документ представляется в виде вектора в n размерном пространстве, в котором каждая размерность соответствует терму из словаря всей коллекции документов.

Другими словами, каждый документ представляется как вектор весов термов, в котором каждый вес указывает на степень связи между документом и термом.

Пусть $D = \{d_1, d_2, \dots, d_N\}$ обозначает набор документов (корпус) и $T = \{t_1, t_2, \dots, t_n\}$ – словарь, сформированный из слов документов D.

Словарь T получаем посредством применения некоторых стандартных операций по обработке естественных языков, таких как токенизация, удаление стоп-слов, стемминг. Каждый документ d_j представляется как вектор в n-размерном векторном пространстве по количеству термов в словаре. Таким образом документ d_j записывается как

$$d_j = \{w_{1j}, w_{2j}, \dots, w_{nj}\},$$

где w_{kj} – вес терма t_k в документе d_j.

Для использования векторного пространства для представления документов необходимо определить:

- Как рассчитывать веса термов
- Как определять сходство между векторами

3.2. Вес термов

В качестве веса термов, как правило, используют TF-IDF (частота термов (TF) – обратная частота документов (IDF)). TF-IDF основано на наблюдениях касаемых текстов:

- Редкие термы не менее релевантны, чем частые термы (IDF)
- Множественное вхождение терма в документ не менее релевантно, чем единственное вхождение (TF)
- Длинные документы не являются предпочтительными по отношению к коротким (нормализация)

Другими словами, термы, которые часто встречаются в одном документ (TF), но редко в остальных (IDF), более вероятно будут релевантными теме документа. Дополнительно нормализация позволяет предотвратить ситуацию, когда более длинные документы имеют больше шансов быть извлеченными (то есть алгоритм поиска скорее выдаст в качестве результата более длинный документ).

TF-IDF рассчитывается следующим образом

$$\mathrm{TFIDF}(t_k, d_j) = \mathrm{TF}(t_k, d_j) \cdot \underbrace{\log \frac{N}{n_k}}_{\mathrm{IDF}},$$

где N – количество документов в коллекции; n_k – количество документов в коллекции, в которых терм t_k встречается по крайней мере один раз.

Для вычисления частоты терма можно использовать нормализованный вариант:

$$\mathrm{TF}(t_k, d_j) = \frac{f_{kj}}{\max\limits_i f_{ij}}$$

где f_{kj} – количество вхождений терма t_k в документ d_j.

Для того, чтобы значения весов лежали в интервале $[0,1]$ и чтобы документы были представлены векторами одной длины, как правило, используют L2 нормализацию:

$$w_{kj} = \frac{\mathrm{TFIDF}(t_k, d_j)}{\sqrt{\sum_{i=1}^{|T|} \mathrm{TFIDF}(t_i, d_j)^2}}$$

3.3. Сходство между векторами

Мера сходства необходима для определения близости между двумя документами. Наиболее часто используемой является косинусное сходство:

$$\mathrm{sim}(d_i, d_j) = \frac{\sum_k w_{ki} \cdot w_{kj}}{\sqrt{\sum_k {w_{ki}}^2} \cdot \sqrt{\sum_k {w_{kj}}^2}}.$$

Замечание

Косинус угла между двумя векторами

$$\cos(\mathbf{x}_a, \mathbf{x}_b) = \frac{\mathbf{x}_a^T \mathbf{x}_b}{\|\mathbf{x}_a\| \|\mathbf{x}_b\|}$$

В рекомендательных системах на основе контента профайлы пользователей и товары представляются в виде векторов весов термов. Для предсказания товаров, которые могут заинтересовать пользователя, вычисляется косинусное сходство между профайлом пользователя (например, на основе описаний из истории покупок) и описанием товаров.

3.4. Достоинства и недостатки

Достоинства:

- Не зависит от других пользователей
- Нет проблемы холодного старта при появлении нового товара
- Интерпретируемость результата
- Может рекомендовать специфичный для конкретного пользователя контент

Недостатки:

- Проблема холодного старта при добавлении нового пользователя
- Выбор подходящих признаков
- В качестве рекомендаций исключается контент, выходящий за рамки истории пользователя (не способен рекомендовать кардинально отличные товары от тех, которые уже были, например, приобретены, но в то же время могли бы понравиться данному пользователю)

4. Коллаборативная фильтрация

4.1. Общие сведения

Коллаборативная фильтрация строится на основе метода ближайших соседей с использованием матрицы рейтингов.

Существуют два подхода:

- на основе схожести пользователей (user-based)

- на основе схожести объектов (item-based)

4.2. Подход на основе схожести пользователей (user-based)

Методы рекомендации на основе схожести пользователей предсказывают рейтинг R_{ui} пользователя u для нового товара i с использованием рейтингов, проставленных товару i наиболее похожими на u пользователями (ближайшие соседи).

Предположим, что для каждого пользователя $u \neq v$ значение s_{uv} представляет сходство между u и v. Для пользователя u k-ближайших соседей, обозначаемое как $N(u)$, есть k пользователей v с наибольшим значением схожести s_{uv}.

В то же время только пользователи, кто проставил рейтинг товару i, могут быть использованы для предсказания R_{ui}. Таким образом, мы должны рассматривать k наиболее похожих пользователей, кто проставил рейтинг товару i. Обозначим множество таких пользователей как $N_i(u)$.

Способы определения сходства между пользователями:

- Косинусное сходство

$$s_{uv} = \text{sim}(u, v) = \frac{\sum_{i \in I} R_{u,i} R_{v,i}}{\sqrt{\sum_{i \in I_u} R_{u,i}^2} \sqrt{\sum_{i \in I_v} R_{v,i}^2}}$$

- Корреляция Пирсона

$$s_{uv} = \text{sim}(u, v) = \frac{\sum_{i \in I} \left(R_{u,i} - \overline{R}_u\right)\left(R_{v,i} - \overline{R}_v\right)}{\sqrt{\sum_{i \in I} \left(R_{u,i} - \overline{R}_u\right)^2} \sqrt{\sum_{i \in I} \left(R_{v,i} - \overline{R}_v\right)^2}}$$

Предсказание

Оценка рейтинга R_{ui} может быть вычислена как среднее значение рейтингов товара i всех соседей:

$$\hat{R}_{ui} = \frac{1}{|N_i(u)|} \sum_{v \in N_i(u)} R_{vi}$$

Данное выражение не учитывает, что соседи имеют разный уровень схожести.

Для решения обозначенной проблемы используется выражение с весами для учета степени схожести:

$$\hat{R}_{ui} = \frac{\sum_{v \in N_i(u)} s_{uv} R_{vi}}{\sum_{v \in N_i(u)} |s_{uv}|}$$

В то же время до сих пор мы не учли, что разные пользователи могут вкладывать разное значение в выставляемые ими рейтинги. Например, некоторые выставляя рейтинг 4 имеют в виду, что товар не понравился, другие наоборот, что товар понравился.

Для решения данной проблемы используется нормализация рейтинга:

- центрирование по среднему
- стандартизация.

Центрирование по среднему:

$$h(R_{ui}) = R_{ui} - \overline{R}_u$$

где $\overline{R}_u$ – средний рейтинг пользователя u товарам в I_u.

Тогда предсказание рейтинга R_{ui} на основе схожести пользователей получим равным

$$\hat{R}_{ui} = \overline{R}_u + \frac{\sum_{v \in N_i(u)} s_{uv}\left(R_{vi} - \overline{R}_v\right)}{\sum_{v \in N_i(u)} |s_{uv}|}$$

Стандартизация

$$h(R_{ui}) = \frac{R_{ui} - \overline{R}_u}{\sigma_u},$$

где σ_u – стандартное отклонение рейтингов пользователя u товарам в I_u.

Тогда предсказание рейтинга R_{ui} на основе схожести пользователей получим равным

$$\hat{R}_{ui} = \overline{R}_u + \sigma_u \frac{\sum_{v \in N_i(u)} s_{uv}\left(R_{vi} - \overline{R}_v\right)/\sigma_v}{\sum_{v \in N_i(u)} |s_{uv}|}$$

4.3. Подход на основе схожести объектов (item-based)

- Пусть пользователь u проставил рейтинги некоторым объектам из $\mathcal{I}$.
- Для некоторого объекта i определяется насколько он похож на объекты пользователя u с проставленными рейтингами
- Выбирается k наиболее похожих объектов $\{i_1, i_2, \dots, i_k\}$. Сходство объектов в этом случае обозначим как $\{s_{i,1}, s_{i,2}, \dots, s_{i,k}\}$.
- Предсказание рейтинга объекта i текущего пользователя u вычисляется как взвешенное среднее рейтингов пользователя на множестве k наиболее похожих объектов.

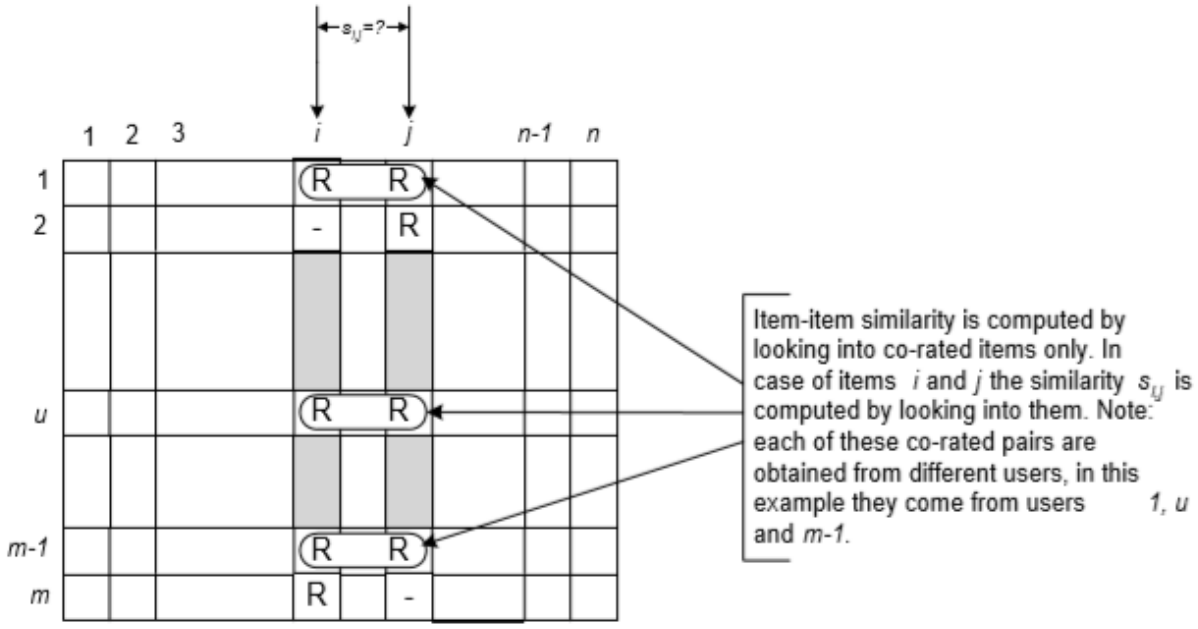


Рисунок – Определение сходства между товарами [2]

Вычисление схожести

Основная идея в определении схожести между объектами i и j заключается в отборе пользователей с проставленными рейтингами для обоих объектов и применение одного из способа вычисления сходства $s_{i,j}$

Способы определения сходства между объектами:

- Косинусное сходство

$$s_{ij} = \text{sim}(i,j) = \frac{\sum_{u \in U} R_{u,i} R_{u,j}}{\sqrt{\sum_{u \in U_i} R_{u,i}^2} \sqrt{\sum_{u \in U_j} R_{u,j}^2}},$$

где $R_{u,i}$ – рейтинг объекта i пользователя u; $U = U_{ij} \subseteq \mathcal{U}$

- Корреляция Пирсона

$$s_{ij} = \text{sim}(i,j) = \frac{\sum_{u \in U} \left(R_{u,i} - \overline{R}_i\right)\left(R_{u,j} - \overline{R}_j\right)}{\sqrt{\sum_{u \in U} \left(R_{u,i} - \overline{R}_i\right)^2} \sqrt{\sum_{u \in U} \left(R_{u,j} - \overline{R}_j\right)^2}},$$

где $\overline{R}_i$ – средний рейтинг объекта i

- Скорректированное косинусное сходство

Корреляция Пирсона учитывает среднее значение рейтинга объектов. Однако в данном случае, когда рассчитывается сходство объектов по столбцам матрицы, правильнее корректировать значения рейтингов по среднему значению рейтинга пользователей.

$$s_{ij} = \text{sim}(i,j) = \frac{\sum_{u \in U}\left(R_{u,i} - \overline{R}_u\right)\left(R_{u,j} - \overline{R}_u\right)}{\sqrt{\sum_{u \in U}\left(R_{u,i} - \overline{R}_u\right)^2}\sqrt{\sum_{u \in U}\left(R_{u,j} - \overline{R}_u\right)^2}},$$

где $\overline{R}_u$ – средний рейтинг объектов пользователя u

Предсказание

Взвешенная сумма

$$P_{u,i} = \hat{R}_{u,i} = \frac{\sum_{j \in \mathcal{N}_u(i)} s_{ij} R_{u,j}}{\sum_{j \in \mathcal{N}_u(i)} |s_{ij}|}$$

$\mathcal{N}_u(i)$ – объекты пользователя u, которые наиболее похожи на объект i.

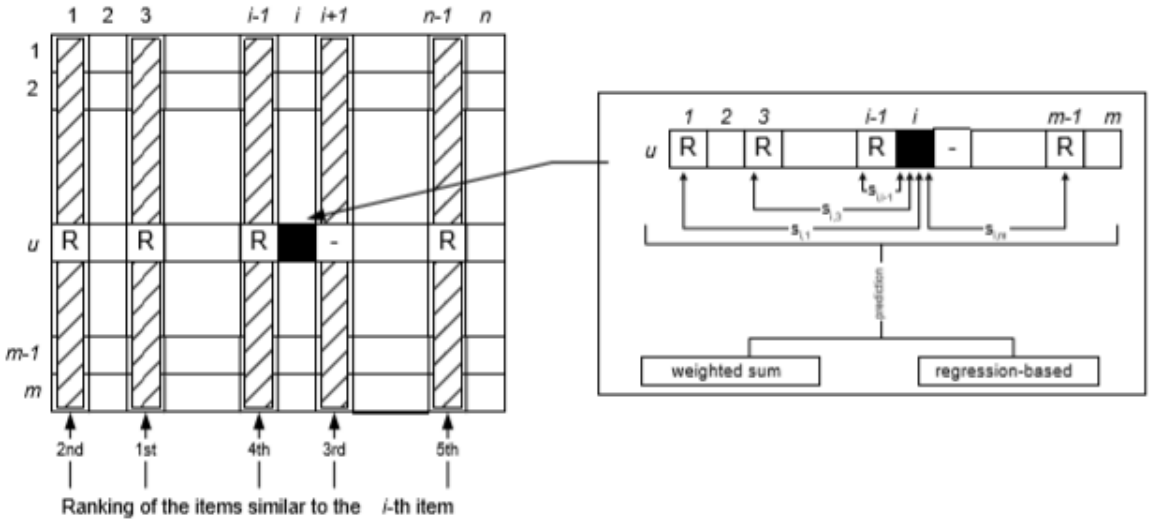


Рисунок – Предсказание рейтинга для товара, которому пользователь не проставил рейтинг [2]

Центрирование по среднему:

$$h(R_{ui}) = R_{ui} - \overline{R}_i$$

где $\overline{R}_i$ – среднее значений рейтингов для товара i, проставленные пользователями из U_i.

Предсказание R_{ui} на основе схожести объектов:

$$\hat{R}_{ui} = \overline{R}_i + \frac{\sum_{j \in N_u(i)} s_{ij}\left(R_{uj} - \overline{R}_j\right)}{\sum_{j \in N_u(u)} |s_{ij}|}$$

Стандартизация:

$$h(R_{ui}) = \frac{R_{ui} - \overline{R}_i}{\sigma_i}$$

Предсказание R_{ui} на основе схожести объектов:

$$\hat{R}_{ui} = \overline{R}_i + \sigma_i \frac{\sum_{j \in N_u(i)} s_{ij} \left(R_{uj} - \overline{R}_j\right)/\sigma_j}{\sum_{j \in N_u(i)} |s_{ij}|}$$

4.4. Сравнение

Качество предсказания

Когда количество пользователей намного больше, чем количества товаров, методы на основе схожести товаров могут дать более точные рекомендации. Пример, Amazon.

В системах с меньшим количеством пользователей по сравнению с количеством товаров рекомендации на основе пользователей могут показать лучший результат. Пример, рекомендации научных статей.

Эффективность

Когда количество пользователей превосходит количество товаров методы рекомендации на основе схожести объектов требуют значительно меньше памяти и времени на вычисление сходства.

Однако время предсказания, которое зависит только от количества доступных товаров и максимального количества соседей, будет одинаковы в обоих подходах.

Стабильность

Выбор между подходом на основе пользователей и на основе объектов также зависит от частоты и количества изменений в пользователях и товарах системы.

Если список доступных товаров достаточно статичный по сравнению с пользователями, то подход на основе объектов может быть предпочтительнее, так как можно нечасто производить вычисление схожести, но при этом по-прежнему быть способным рекомендовать товары новым пользователям.

Интерпретация предсказаний

В методах на основе схожести объектов предсказание рейтинга товара вычисляется по рейтингам похожих товаров. Следовательно, рекомендательные системы, использующие данный подход, будут иметь тенденцию рекомендовать товары, связанные с теми, которые до этого были высоко оценены пользователем. Например, при рекомендации фильмов те из них, которые имеют одинаковый жанр, актеров, режиссера с теми фильмами, что получили высокий рейтинг от данного пользователя, скорее всего будут ему рекомендованы.

С одной стороны, это позволяет получить достаточно адекватный результат предсказания, однако с другой стороны не позволяет пользователю открыть для себя что-то новое.

4.5. Достоинства и недостатки

Достоинства:

- Нет необходимости выбирать признаки, так как использоваться только матрица рейтингов

Недостатки:

- Проблема холодного старта, так как требуется достаточное количество пользователей с рейтингами
- Проблема холодного старта при добавлении новых товаров
- Матрица рейтингов разрежена
- Имеет тенденцию рекомендовать популярные товары

5. Факторизация матрицы рейтингов

5.1. Общие сведения

Основная задача – это разложить матрицу рейтингов R на произведение двух матриц W и H. При этом предсказание рейтинга будет вычисляться как произведение строки матрица W на столбец матрицы H. Количество факторов, которые определяют скрытые признаки, контролирует полноту разложения. Наша задачи уменьшить количество факторов, но при этом сохранить качество предсказания.

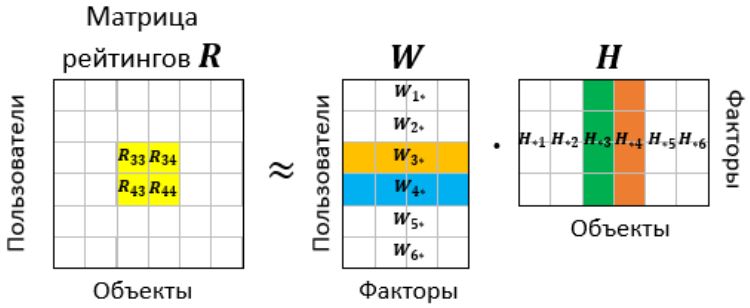


Рисунок – Разложение матрицы рейтингов на матрицы пользователей и товаров

Задача оптимизации

$$\widehat{\boldsymbol{W}}, \widehat{\boldsymbol{H}} = \arg\min_{\boldsymbol{W},\boldsymbol{H}} L(\boldsymbol{W}, \boldsymbol{H})$$

Функция потерь

$$L(\boldsymbol{W}, \boldsymbol{H}) = \sum_{(i,j)\in\Omega} L_{ij}(\boldsymbol{W}, \boldsymbol{H})$$

$$L_{ij}(\boldsymbol{W}, \boldsymbol{H}) = l\left(\boldsymbol{R}_{ij}, \boldsymbol{W}_{i*}, \boldsymbol{H}_{*j}\right)$$

Пример функции потерь (без регуляризации)

$$L(\boldsymbol{W}, \boldsymbol{H}) = \sum_{(i,j)\in\Omega} \left(\boldsymbol{R}_{ij} - \boldsymbol{W}_{i*}\boldsymbol{H}_{*j}\right)^2$$

Факторизация неполной матрицы:

- Градиентный спуск
- Стохастический градиентный спуск
- Чередование наименьших квадратов (Alternating Least Squares – ALS)

5.2. Метод наименьших квадратов с чередованием

Функция потерь для задачи регрессии:

$$L(\boldsymbol{W}, \boldsymbol{H}) = \sum_{(i,j)\in\Omega} \left(\boldsymbol{R}_{ij} - \boldsymbol{W}_{i*}\boldsymbol{H}_{*j}\right)^2$$

Частные производные по $\boldsymbol{W}_{i*}$:

$$\frac{\partial L(\boldsymbol{W}, \boldsymbol{H})}{\partial \boldsymbol{W}_{i*}} = 0$$

Частные производные по $\boldsymbol{H}_{*j}$:

$$\frac{\partial L(\boldsymbol{W}, \boldsymbol{H})}{\partial \boldsymbol{H}_{*j}} = 0$$

Приравниваем частные производные к нулю для вычисления параметров модели:

$$\boldsymbol{R}_{i*} - \boldsymbol{W}_{i*}\boldsymbol{H}^{(n)} = \boldsymbol{0}$$

$$\boldsymbol{R}_{*j} - \boldsymbol{W}^{(n+1)}\boldsymbol{H}_{*j} = \boldsymbol{0}$$

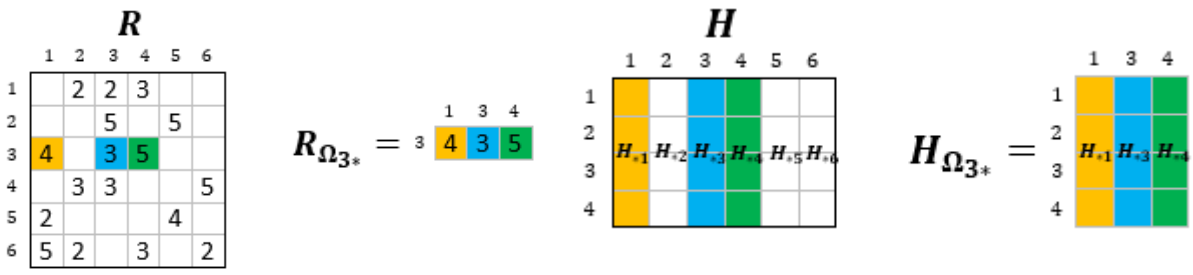
Шаг 1. Вычисление $\boldsymbol{W}$ при фиксированном $\boldsymbol{H}$

Метод наименьших квадратов для задачи линейной регрессии:

$$\widehat{\boldsymbol{\theta}} = \left(\mathrm{X}^{\mathrm{T}}\mathrm{X}\right)^{-1}\mathrm{X}^{\mathrm{T}}\mathbf{y}$$

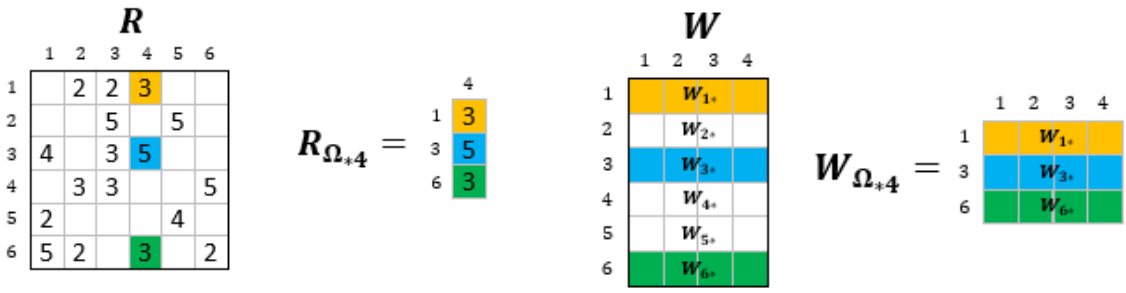
Для матрицы рейтингов

$$\boldsymbol{W}_{i*}^{(n+1)^{\boldsymbol{T}}} = \left(\boldsymbol{H}_{\boldsymbol{\Omega}_{i*}}^{(n)}\boldsymbol{H}_{\boldsymbol{\Omega}_{i*}}^{(n)^{\boldsymbol{T}}}\right)^{-\boldsymbol{1}} \boldsymbol{H}_{\boldsymbol{\Omega}_{i*}}^{(n)}\boldsymbol{R}_{\boldsymbol{\Omega}_{i*}}{}^{\boldsymbol{T}}$$



Шаг 2. Вычисление $\boldsymbol{H}$ при фиксированном $\boldsymbol{W}$

$$\boldsymbol{H}_{*j}^{(n+1)} = \left(\boldsymbol{W}_{\Omega_{*j}}^{(n+1)^T} \boldsymbol{W}_{\Omega_{*j}}^{(n+1)}\right)^{-1} \boldsymbol{W}_{\Omega_{*j}}^{(n+1)^T} \boldsymbol{R}_{\Omega_{*j}}$$



Алгоритм

1 $\boldsymbol{H} \leftarrow$ средний рейтинг для первой строки и малые случайные значения для остальных
2 Цикл: критерий остановки
3 $\quad \boldsymbol{W} \leftarrow$ вычисление при фиксированном $\boldsymbol{H}$
4 $\quad \boldsymbol{H} \leftarrow$ вычисление при фиксированном $\boldsymbol{W}$

Список литературы

1. Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. 2010. Recommender Systems Handbook (1st. ed.). Springer-Verlag, Berlin, Heidelberg
2. Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web (WWW '01). Association for Computing Machinery, New York, NY, USA, 285–295. DOI:https://doi.org/10.1145/371920.372071
3. István Pilászy, Dávid Zibriczky, and Domonkos Tikk. Fast als-based matrix factorization for explicit and implicit feedback datasets. In Proceedings of the fourth ACM conference on Recommender systems (RecSys '10). Association for Computing Machinery, New York, NY, USA, 71–78. DOI:https://doi.org/10.1145/1864708.1864726