Content-based Model

Подход **content-based** предполагает, что пользователю рекомендуются товары или контент на основе его предпочтений и вкусов.

Индекс Жаккара

Индекс Жаккара измеряет сходство между двумя наборами *A* и *B* как мощность (количество объектов) множества пересечения, делённую на мощность множества объединения каких-то характеристик объекта. Его удобно применять для категориальных признаков.

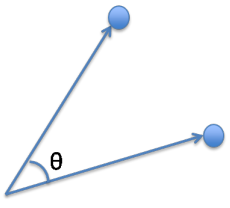
$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Косинусная близость

Подход с косинусным сходством немного сложнее и применяется для оценки близости массивов с числами. Он требует, чтобы мы представляли объекты в виде вектора.

Чтобы вычислить косинусную близость, нам понадобится следующая формула:

$$sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$



По сути, мы ищем угол между векторами. Если два вектора совпадут (т. е. будут максимально близкими), то угол между ними будет равен нулю, а значит, косинус будет равен 1. Если векторы будут направлены в противоположные стороны, косинус будет равен -1. Таким образом, мы можем получить для любых двух векторов значение в пределах от -1 до 1 включительно, по которому можно определить, насколько векторы близки друг к другу.

Чтобы построить рекомендательную систему на основе контента, необходимо:

- ➔ Для каждого продукта создать характеризующие его признаки.
- ➔ Найти показатель близости между всеми продуктами.
- ➔ Порекомендовать пользователю продукты, которые показывают наибольшую близость с теми продуктами, которые он высоко оценил.

> **Показатель TD-IDF** — это индикатор того, насколько релевантно слово в контексте документа.

Его можно определить следующим образом:

$$\text{TF}-\text{IDF}(\text{слова}) = \text{TF}(\text{слова}) * \text{IDF}\,(\text{слова})$$, где:

- **TF** слова = $\frac{\text{количество раз, когда слово встретилось в тексте}}{\text{количество всех слов в тексте}}$;
- **IDF** слова = $log(\frac{\text{общее количество документов}}{\text{количество документов, в которых встречается слово}})$.

Этот показатель возрастает **пропорционально** количеству раз, когда слово встречается в тексте, и уменьшается пропорционально количеству слов во всех текстах в целом.

Таким образом:

- ➔ Коэффициент будет выше, если слово характерно именно для этого текста, то есть встречается в данном тексте часто, но не встречается в других текстах.
- ➔ Коэффициент будет ниже, если слово не встречается почти нигде или встречается одинаковое количество раз во всех текстах, то есть не характеризует никакой текст в отдельности.

Преимущества и недостатки content-based model

- Для создания рекомендаций не требуются данные от других пользователей.
- Рекомендации получаются очень релевантными для пользователя.
- Вы избегаете проблемы «холодного старта».
- Системы фильтрации на основе содержания обычно проще в создании.

- Отсутствие новизны и разнообразия.
- Присвоенные характеристики могут быть неверными.

Коллаборативная фильтрация

В целом, такой подход можно применять, однако у него есть ряд существенных недостатков:

- ➔ Нечего рекомендовать новым/нетипичным пользователям.
- ➔ Не учитывается специфика каждого пользователя.
- ➔ Если в кластере никто не оценивал объект, сделать предсказание не получится, так как для предсказания нужно вычислить среднее арифметическое для оценок.

Коллаборативная фильтрация на основе памяти (memory-based)

При *memory-based*-подходе хранится полная матрица взаимодействий (лайков, просмотров и т. д .) пользователя с продуктом.

Коллаборативная фильтрация на основе пользователей. User-based-подход

Коллаборативная фильтрация на основе пользователей — это метод, используемый для предсказания продуктов, которые могут понравиться пользователю, на основе оценок, выставленных этому продукту другими пользователями, имеющими схожие с целевым пользователем вкусы.

В этом алгоритме мы заменяем жёсткую кластеризацию на следующую формулу и получаем предсказанную оценку пользователя u, которую он поставил для элемента i:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{v \in U_i} \text{sim}(u, v)(r_{vi} - \bar{r}_v)}{\sum_{v \in U_i} \text{sim}(u, v)}$$

, где

- u и v — индексы пользователей;
- $\overline{r_u}$ — средняя оценка пользователя u;
- $\overline{r_v}$ — средняя оценка пользователя v;
- sim — функция схожести;
- i — номер оцениваемого элемента.

Каждому клиенту мы подбираем релевантный для него товар в рамках группы клиентов, но не решаем задачу кластеризации, а усредняем интересы данной группы в дистанции нескольких соседей.

Коллаборативная фильтрация на основе элементов. Item-based-подход

Если мы транспонируем матрицу предпочтений и будем решать ту же самую задачу не для пользователей, а для объектов *(items)*, то получим аналогичную задачу, которая является *item-based*-моделью коллаборативной фильтрации и даёт нам возможность предсказывать оценку следующим образом:

$$\hat{r}_{ui} = \bar{r}_i + \frac{\sum_{j \in I_u} \text{sim}(i,j)(r_{uj} - \bar{r}_j)}{\sum_{j \in I_u} \text{sim}(i,j)}$$

Может показаться, что коллаборативная фильтрация в рамках *item-based*-подхода очень похожа на модель на основе контента. Однако это не так: *item-based*-модель рассматривает взаимодействия пользователей с продуктом, а *content-based*-модель — метаинформацию продукта.

Преимущества и недостатки коллаборативной фильтрации, основанной на памяти (memory-based):

- Может помочь пользователям обнаружить новые релевантные продукты из новых категорий, даже если пользователи не проявляют интерес к новым типам продуктов.
- Не требует подробных характеристик и контекстных данных о продуктах. По сути, для реализации нужна только матрица взаимодействий пользователей и продуктов.

- Нехватка данных может привести к трудностям при рекомендации новых продуктов или при появлении новых пользователей, поскольку предложения основаны на исторических данных и взаимодействии.

Таким образом, для этого подхода актуальна проблема холодного старта.

- ➔ По мере роста пользовательской базы алгоритмы страдают из-за большого объёма данных (очень ресурсоёмкие вычисления) и недостаточной масштабируемости.
- ➔ Отсутствие разнообразия в долгосрочной перспективе.
  Это может показаться нелогичным, поскольку смысл коллаборативной фильтрации заключается в том, чтобы рекомендовать пользователю новые товары. Однако поскольку алгоритмы функционируют на основе исторических рейтингов, они не будут рекомендовать товары с небольшим количеством оценок или ограниченным количеством данных. Популярные товары будут более популярны в долгосрочной перспективе, а новых и разнообразных вариантов будет не хватать.

Model-based-подход

В подходе на основе моделей используются модели машинного обучения для прогнозирования и ранжирования взаимодействий между пользователями и элементами, с которыми они ещё не взаимодействовали. Эти модели обучаются на основе информации, уже имеющейся в матрице взаимодействий, с помощью различных алгоритмов, например матричной факторизации.

Матричная факторизация используется для генерации **латентных признаков** путём разложения **разрежённой** матрицы взаимодействия пользователя и продукта на две меньшие и **плотные** матрицы особенностей пользователей и продуктов.

SVD — это сингулярное разложение.

Любую прямоугольную матрицу *A* размера *(n, m)* можно представить в виде произведения трёх матриц:

$$A_{n\times m} = U_{n\times n} \cdot D_{n\times m} \cdot V^T_{m\times m}$$

- U — матрица размера *(n, n)*. Все её столбцы ортогональны друг другу и имеют единичную длину. Такие матрицы называются **ортогональными**. Эта матрица содержит нормированные собственные векторы матрицы AA^T.
- U — матрица размера *(n, m)*. На её главной диагонали стоят числа, называемые **сингулярными** числами (они являются корнями из собственных значений матриц AA^T и A^TA), а вне главной диагонали стоят нули. Если мы решаем задачу снижения размерности, то элементы этой матрицы, если их возвести в квадрат, можно интерпретировать как дисперсию, которую объясняет каждая компонента.
- V — матрица размера *(m, m)*. Она тоже **ортогональная** и содержит нормированные собственные векторы матрицы A^TA.

Данное разложение используют для того, чтобы представить матрицу предпочтений как разложение на матрицу с характеристиками пользователей и матрицу с характеристиками продуктов.

ALS — итеративный алгоритм разложения матрицы предпочтений на произведение двух матриц. Чтобы понять лучше суть этого алгоритма, вспомним одну из известных нам функций потерь — *RMSE*.

$$RMSE = \sqrt{(y - \hat{y})^2/n}$$

Предположим, что есть m пользователей и n продуктов. Тогда у нас будут следующие матрицы:

- R размерности m * n;
- U размерности m * k;
- P размерности n * k, где k — количество латентных факторов.

Тогда функция потерь, которую мы будем минимизировать, будет следующей:

$$\begin{aligned} \text{loss} &= \min(y - \hat{y})^2 \\ &= \min\left(R - U * P^T\right)^2 \\ &= \min \sum_{x,y} \left(R_{m,n} - U_m * P_n^T\right)^2 \end{aligned}$$

Здесь:

- R — истинные показатели взаимодействия пользователя и продукта;
- $U * P^T$ — прогнозируемые показатели взаимодействия пользователя и продукта.

ALS — это итерационный процесс оптимизации, в котором мы на каждой итерации пытаемся приблизиться к факторизованному представлению исходных данных.

Гибридные модели

Гибридная РС — это особый тип рекомендательной системы, который представляет собой комбинацию из нескольких методов. Обычно это комбинация контентного подхода и коллаборативной фильтрации. Такое сочетание может помочь преодолеть недостатки, с которыми мы сталкиваемся при использовании этих методов по отдельности, а также в некоторых случаях может быть более эффективным.

Разумеется, можно комбинировать различные подходы самостоятельно, однако для удобства уже реализован модуль *LightFM*.

В модуле *LightFM* представлены следующие функции потерь:

- `'logistic'` — логистическая функция. Полезна в случаях, когда есть как положительные, так и отрицательные взаимодействия, например 1 и -1.
- `'bpr'` — байесовский персонализированный рейтинг. Можно применять, когда присутствуют только положительные взаимодействия.

- `'warp'` — парный взвешенный приблизительный ранг. Используется, если необходимо повысить качество именно в верхней части списка рекомендаций.
- `'warp-kos'` — модификация `warp`.

Преимущества и недостатки

- Использование разных моделей позволяет компенсировать их недостатки и использовать преимущества каждой модели для составления рекомендаций.
- Благодаря использованию сразу нескольких методов такой подход позволяет получить более персонализированные и точные рекомендации.

- Такие модели обычно имеют высокую вычислительную сложность и из-за этого долго обучаются.
- Для обучения таких моделей требуются большие объёмы данных и поступление новых данных для обновления рекомендаций.

Современные методы: глубокое обучение

Преимущества использования нейронных сетей:

➔ Как правило, *DL*-модели дают более высокое качество.
➔ *DL*-модели обладают большей гибкостью.
➔ Можно включать в модель данные совершенно разных типов.

Эмбеддинг — это пространство низкой размерности, которое отражает взаимосвязь векторов из пространства более высокой размерности.

По сути, эмбеддинги уменьшают размерность данных, оставляя осмысленность в их отображении. При таком преобразовании пользователи, предпочтения которых похожи, находятся рядом в плоскости (или пространстве), а пользователи, предпочтения которых отличаются, находятся далеко друг от друга.

Необязательно использовать для формирования эмбеддинга только два числа — для вектора можно взять любое количество компонент. Причём чем больше будет измерений, тем выше будет точность представления всех особенностей пользователя и, как следствие, выше будет точность модели. Однако следует понимать, что такое увеличение точности происходит за счёт роста сложности модели и увеличения количества времени, необходимого для её обучения.

Обычно для улучшения качества модели каким-то образом модифицируют нейронную сеть: дополняют её, увеличивают время обучения.