

Машинное обучение
Лекция №10, осень 2021

Рекомендательные системы



План лекции

- Постановка задачи
- Неперсонализированные рекомендации
- Подходы к построению рекомендательных систем
 - Collaborative filtering
 - Content-based
- Оценка качества
 - offline
 - online

Постановка задачи

U - множество субъектов (users/пользователи/клиенты);

I - множество объектов (items/предметы/товары/ресурсы);

Y - пространство описаний транзакций (операций,

$$D = (u_t^M, i_t^K, y_t^L) \in U \times I \times Y$$

- транзакционные данные;

u - векторное описание субъекта размерности M ;

i - векторное описание объекта размерности K ;

y - векторное описание транзакций (вообще говоря многомерное).

Задачи:

- прогнозирование Y ;
- оценивание сходства: $p(u, u')$, $p(i, i')$;
- оценка вероятности транзакции и ее параметров $p(u, i)$;
- формирование списка рекомендаций для u или для i .

Пример 1. Рекомендательная система для e-commerce

U - клиенты интернет магазина;

I - товары;

r_{ui} = [клиент u купил товар i].

Задачи персонализации предложений:

- выдать оценку товара i для клиента u ;
- выдать клиенту u список рекомендуемых товаров;
- предложить совместную покупку (cross-selling);
- информировать клиента о новом товаре (up-selling);
- сегментировать клиентскую базу;
- выделить интересы клиентов (найти целевые аудитории).

Пример 2. Рекомендательная система для web-страниц

U - пользователи интернета;

I - страницы (сайты, документы, новости ...);

r_{ui} = [пользователь u посетил страницу i].

Основная гипотеза Web Usage Mining:

Посещения пользователя характеризуют его интересы, вкусы, привычки и возможности.

Задачи персонализации предложений:

- для пользователя u :
 - выдать оценку страницы i ;
 - выдать ранжированный список рекомендуемых страниц;
- для страницы i : выдать список страниц, близких к i .

Пример 3. Рекомендательная система на основе рейтингов

U - клиенты интернет-магазина;

I - товары (книги, музыка, видео ...);

r_{ui} = рейтинг, который клиент u выставил товару i .

Задачи персонализации предложений те же.

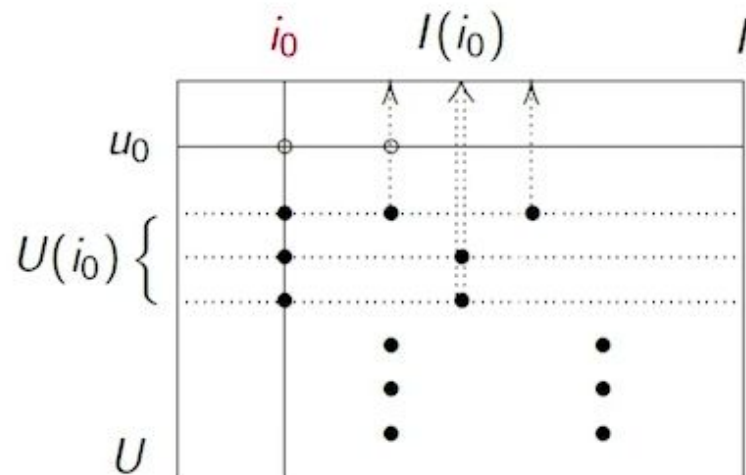
Пример: конкурс [Netflix Prize](#)

- 2 октября 2006 - 21 сентября 2009;
- главный приз - 1 млн \$;
- $|U| = 0,48$ млн человек, $|I| = 17$ млн фильмов;
- точность прогнозов оценивается по RMSE на тестовой выборке;
- задача: уменьшить RMSE с 0,9514 до 0,8563 (на 10%).

Неперсонализированные рекомендации

Не пытается рекомендовать наиболее подходящее конкретному человеку

“Клиенты, купившие A,
также выбирают B”



“Хит продаж”

Рекомендуем самые популярные товары

Отсортировать I по частоте транзакций

$U(i_0) := \{u \in U \mid r_{ui_0} \neq \emptyset, u \neq u_0\}$ — коллаборация;

$I(i_0) := \left\{ i \in I \mid \text{sim}(i, i_0) = \frac{|U(i_0) \cap U(i)|}{|U(i_0) \cup U(i)|} > \delta \right\},$

Неперсонализированные рекомендации

Недостатки:

- рекомендации тривиальны
(предлагается все наиболее популярное);
- не учитываются интересы конкретного пользователя;
- проблема холодного старта
(новый товар никому не рекомендуется);
- надо хранить все данные, в добавок к статистике.

Подходы к построению рекомендательных систем

- Collaborative filtering (*На основе оценок похожих пользователей*)
- Content-based (*На основе признаков для пользователей и объектов*)
- [Demographic](#) (*Классификация пользователей по группам*)
- [Utility-based](#) (*На основе функции полезности - user based utility function*)
- Knowledge-based (*На основе базы знаний о соотношении объектов с интересами пользователей*)

Подходы к построению рекомендательных систем

По типу данных:

Explicit data

Явные данные - это данные, по которым у нас есть своего рода рейтинг. Для таких данных мы знаем, насколько пользователю нравится или не нравится конкретный товар, но такие данные трудно собрать.

Пример: Оценки на Netflix или Яндекс. Музыка (пусть и бинарные).

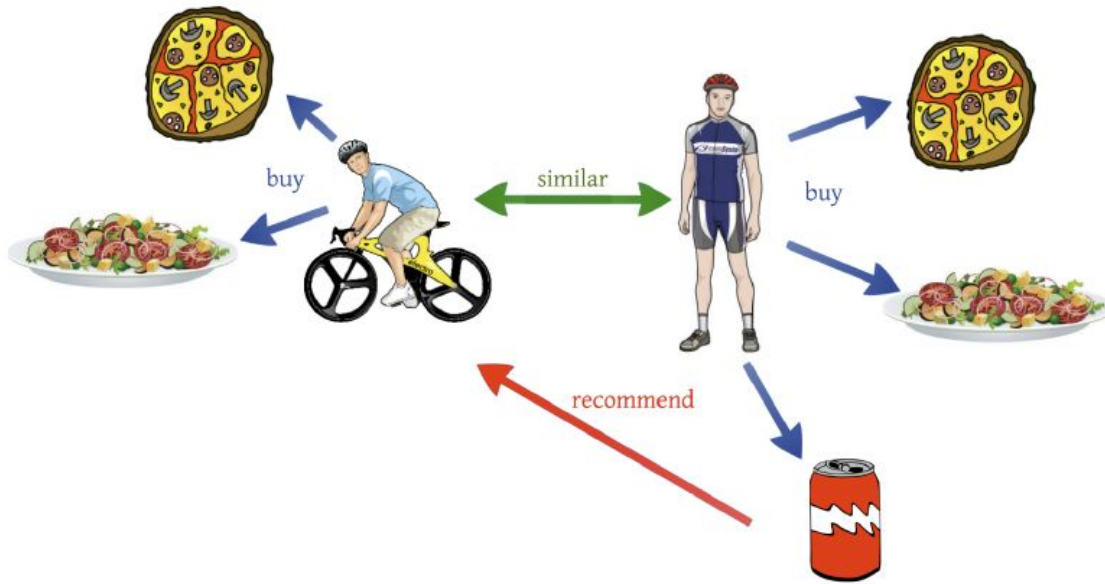
Implicit data






















Неявные данные - это данные, которые мы собираем по поведению пользователей, без каких-либо оценок или конкретных действий. Таких данных гораздо больше, но они более шумные и их сложнее интерпретировать.

Примеры: Какие предметы купил пользователь, сколько раз они слушал песню или смотрели фильм, сколько времени он потратил на чтение конкретной книги.

Collaborative filtering

- User-based
- Item-based



					
A					
B					
C					
D					
E					

Collaborative filtering. Меры близости

корреляция Пирсона:

$$\text{sim}(u, v) = \frac{\sum_{i \in I(u, v)} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I(u, v)} (r_{ui} - \bar{r}_u)^2 \sum_{i \in I(u, v)} (r_{vi} - \bar{r}_v)^2}};$$

косинусная мера близости:

$$\text{sim}(u, v) = \frac{\sum_{i \in I(u, v)} r_{ui} r_{vi}}{\sqrt{\sum_{i \in I(u, v)} r_{ui}^2 \sum_{i \in I(u, v)} r_{vi}^2}};$$

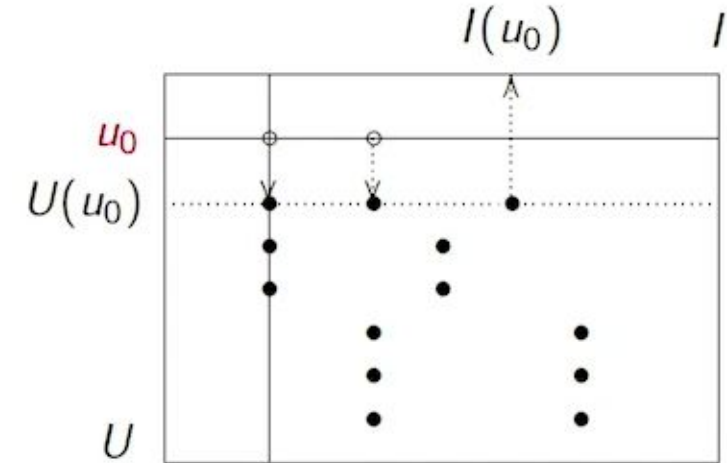
где $I(u, v) = \begin{cases} I(u) \cup I(v), & \text{для бинарных данных,} \\ I(u) \cap I(v), & \text{для рейтинговых данных.} \end{cases}$

* статистические критерии.

Collaborative filtering. User-based

Клиенты, похожие на u , также покупают

$U(u_0) := \{u \in U \mid \text{sim}(u_0, u) > \alpha\}$ — коллаборация;
 $\text{sim}(u_0, u)$ — одна из возможных мер близости u к u_0 ;
 $I(u_0) := \left\{ i \in I \mid B(i) = \frac{|U(u_0) \cap U(i)|}{|U(u_0) \cup U(i)|} > 0 \right\}$;
где $U(i) := \{u \in U \mid r_{ui} \neq \emptyset\}$;



Collaborative filtering. User-based

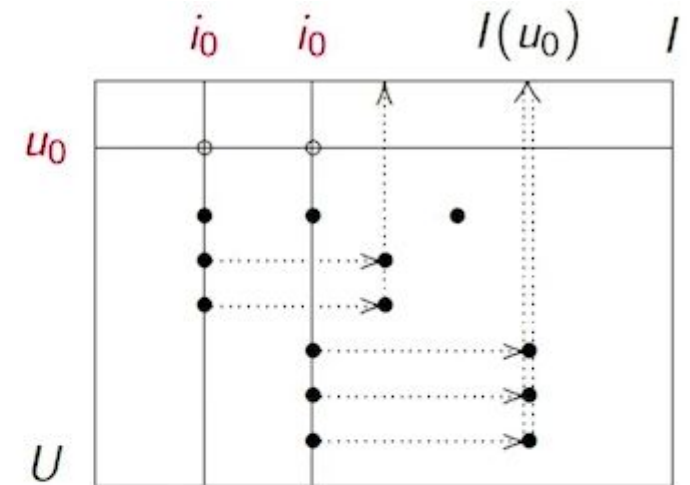
Недостатки:

- рекомендации тривиальны;
- не учитываются интересы конкретного пользователя;
- надо хранить всю матрицу R ;
- нечего рекомендовать нетипичным новым пользователям.

Collaborative filtering. Item-based

Вместе с товарами, которые покупал u_0 , часто покупают $I(u_0)$

$I(u_0) := \{i \in I \mid \exists i_0: r_{u_0 i_0} \neq \emptyset \text{ и } B(i) = \text{sim}(i, i_0) > \alpha\};$
где $\text{sim}(i, i_0)$ — одна из возможных мер сходства i и i_0 ;
сортировка $i \in I(u_0)$ по убыванию $B(i)$, взять top N ;



Collaborative filtering. Item-based

Недостатки:

- рекомендации часто тривиальны (нет коллаборативности);
- проблема «холодного старта»;
- надо хранить всю матрицу R ;
- ~~нечего рекомендовать нетипичным пользователям.~~

Collaborative filtering. Рекомендации

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

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

\bar{r}_u и \bar{r}_i — средний рейтинг клиента u и объекта i ,
 $\text{sim}(u, v)$ и $\text{sim}(i, j)$ — функции близости (u, v) и (i, j) ,

Collaborative filtering. Matrix Factorization

Идея Матричного разложения состоит в том, чтобы перейти от матрица Users*Items к некому представлению “предпочтений” пользователей и “характеристик” объектов.

	item 1	item 2	item 3	...	item n
user 1					
user 2					
user 3					
user 4					
user 5					
user 6					
user 7					
user 8					
...					
user n					

\underline{R}

\approx

	feature 1	feature 2
user 1		
user 2		
user 3		
user 4		
user 5		
user 6		
user 7		
user 8		
...		
user n		

\underline{U}

\times

	item 1	item 2	item 3	...	item n
feature 1					
feature 2					

\underline{V}

Collaborative filtering. Matrix Factorization

- Рекомендации основаны на истории оценок как самого пользователя, так и других.
- Идея состоит в том, чтобы взять большую матрицу и разложить ее на меньшее представление исходной матрицы. В итоге получается две или более матриц меньшей размерности, произведение которых равно исходной.

Collaborative filtering. Matrix Factorization

Реализации

Alternating least squares (ALS):

Цель состоит в том, чтобы найти вектор для каждого пользователя и элемента, для этого мы хотим минимизировать следующую функцию потерь

$$\min_{y_*, y_*} \sum_{u,i} c_{ui} (p_{ui} - x_u^T y_i)^2 + \lambda \left(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2 \right)$$

Беря производные по параметрам, находим значения, которые минимизируют функционал:

$$x_u = (Y^T C^u Y + \lambda I)^{-1} Y^T C^u p(u)$$

$$y_i = (X^T C^i X + \lambda I)^{-1} X^T C^i p(i)$$

Collaborative filtering. Matrix Factorization

Реализации

Наивный метод Такера:

initialize via HOSVD

while not converged do

 for $k = 1, 2, \dots, N$

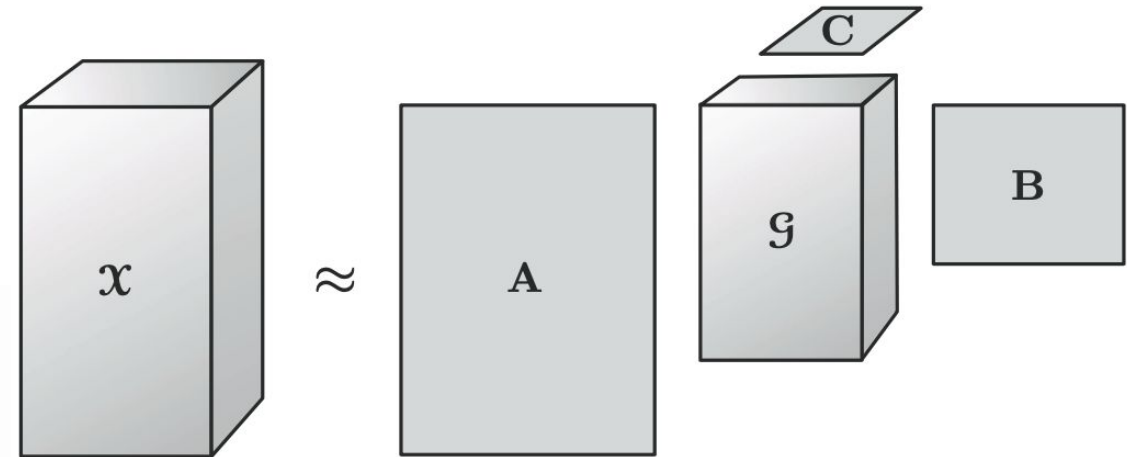
$$Y \leftarrow X \times_1 (A^{(1)})^T \times_2 \cdots \times_{k-1} (A^{(k-1)})^T \times_{k+1} (A^{(k+1)})^T \times_{k+2} \cdots \times_N (A^{(N)})^T$$

$A^{(k)} \leftarrow r_k$ leading left singular vectors of $Y_{(k)}$

 end for

end while

$$G \leftarrow X \times_1 (A^{(1)})^T \times_2 (A^{(2)})^T \times_3 \cdots \times_N (A^{(N)})^T$$

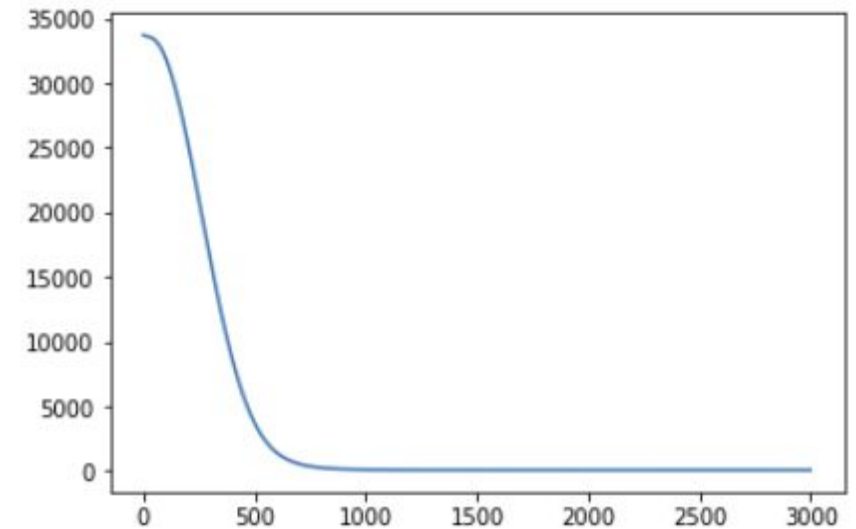


Collaborative filtering. Matrix Factorization

Реализации

Градиентные методы:

- Градиентный
- Пакетный
- Стохастический
- С моментом
- [Alternating Randomized Block Coordinate Descent](#) (ARBCD)



Collaborative filtering. Преимущества и недостатки

Преимущества:

- Хорошо интерпретируется;
- Легко реализуется;
- Дает хорошие результаты (с учетом устранения недостатков).

Недостатки:

- Слабо теоретически обоснован (эвристический алгоритм);
- Все методы требуют хранения и обновления матрицы транзакций;
- Проблема холодного старта.

Content-based

Личная информация о пользователях опускается. Товары и услуги рекомендуются на основе знаний о них:

- жанр,
- производитель,
- конкретные функции
- любые данные, которые можно собрать.

	name	genre
1	Toy Story (1995)	Animation Children's Comedy
2	Jumanji (1995)	Adventure Children's Fantasy
3	Grumpier Old Men (1995)	Comedy Romance
4	Waiting to Exhale (1995)	Comedy Drama
5	Father of the Bride Part II (1995)	Comedy

Преимущества:

- Решение проблемы холодного старта для пользователей, которые дали о себе информацию.
- Не требуется хранения матрицы транзакций.

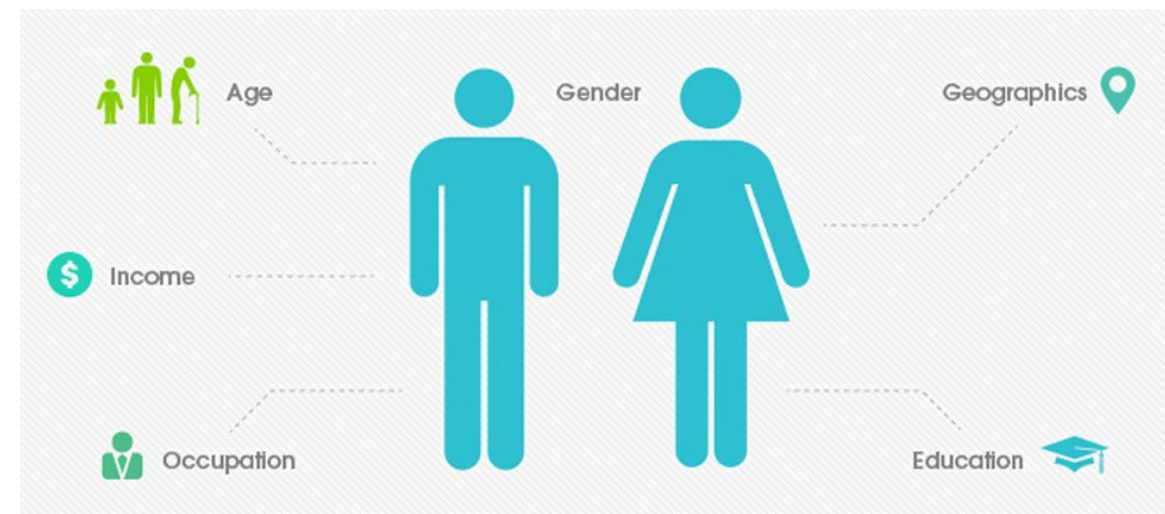
Недостатки:

- Проблема холодного старта, для пользователей, о которых ничего не известно.
- Требуется больше времени на реализацию;
- Ошибки в данных существенно влияют на результаты.

Demographic-based

Demographic-based Recommender рекомендует товары на основе демографической информации пользователей:

- пол;
- возраст;
- место проживания;
- достаток;
- ...



Не требует оценок пользователей или знаний о предметах.

Работает, например, на основе статистики.

Knowledge-based

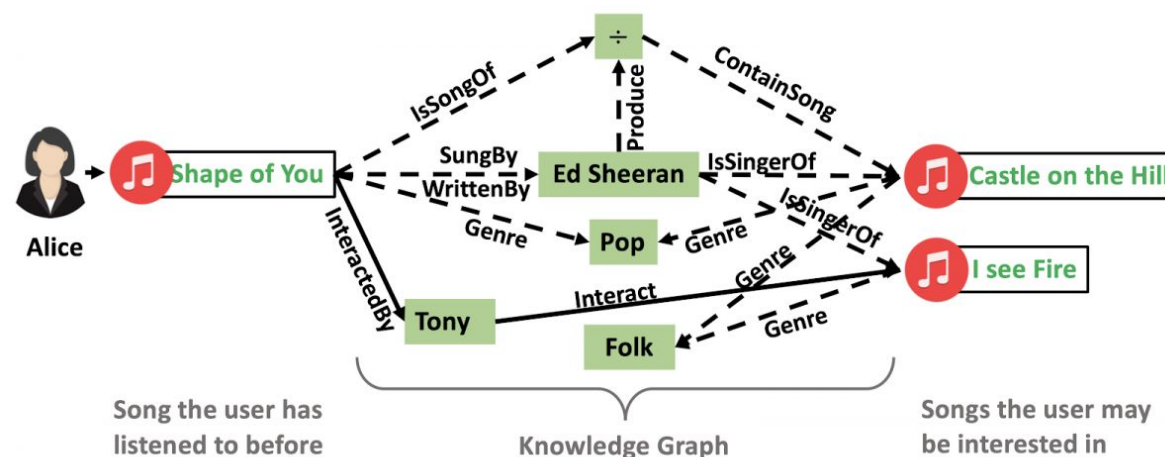
Knowledge-based системы, основаны на явных знаниях об ассортименте товаров, предпочтениях пользователей и критериях рекомендаций (т. е. какой товар должен быть рекомендован в каких случаях).

Основным преимуществом является:

- отсутствие проблемы холодного старта.

Основные недостатки:

- сложность получения знаний;
- сложность построения правил рекомендаций.



Например, при выборе новой игровой консоли PS4 посетителю сайта предлагают купить дополнительные геймпады, шлем виртуальной реальности, популярные игры и другие сопутствующие товары. Как итог, человек совершает больше покупок и увеличивает прибыль компании.

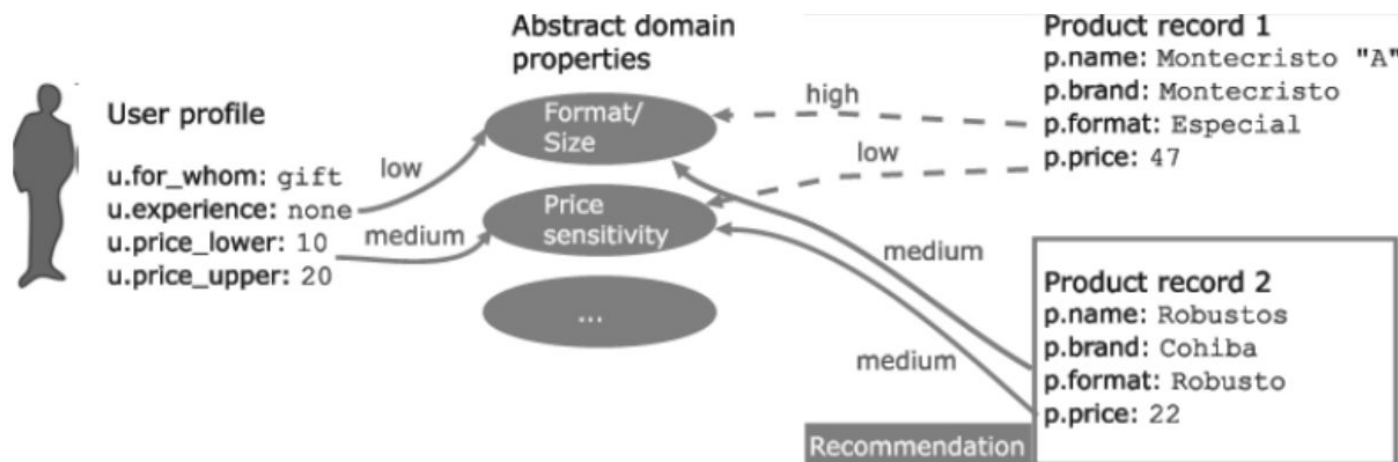
Utility-based

Utility (полезность): мера того, насколько полезна рекомендация для пользователя.

Такие системы строятся на модели пользователя с целью оптимизации его многокритериального выбора из множества вариантов (von Winterfeldt, D. and

W. Edwards: 1986, Decision Analysis and Behavioral Research. Cambridge, UK: Cambridge University Press).

Набор явных пользовательских требований составляет пользовательскую модель и представлен парами атрибут-значение.



Гибридные рекомендательные системы

Типы гибридных рекомендательных систем

- Weighted (Рекомендации строятся на основе комбинирования оценок от разных систем с весами)
- Switching (Мета-алгоритм на основе данных выбирает одну из систем)
- Mixed (Список рекомендаций состоит из “смеси” рекомендаций от разных систем)
- Feature combination (Объединение признаков от разных систем в единую выборку)
- Cascade (Поэтапное применение нескольких моделей) - Candidate selection
- Feature augmentation (Выход одной или нескольких рекомендательных систем используются как входные признаки для другой системы)

Оценка качества

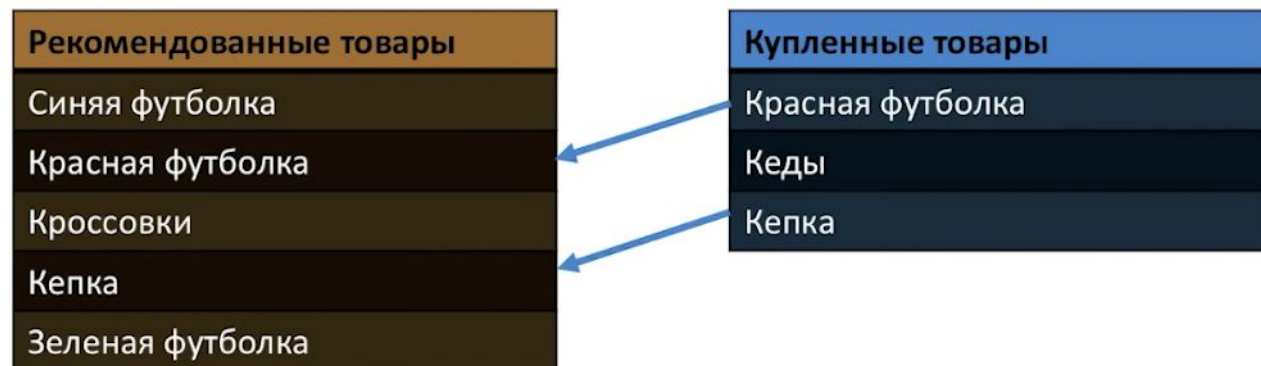
Качество модели = качество прогноза оценок?

- Среднеквадратичное отклонение (RMSE)
- Среднее абсолютное отклонение (MAE)

Что мы оцениваем?

А что нужно оценивать?

Оценка качества. Offline



По аналогии с задачей классификации мы можем использовать метрики Precision и Recall для первых k рекомендаций.

$\text{Precision@k} = \text{куплено из рекомендованного} / k$;

$\text{Recall@k} = \text{куплено из рекомендованного} / \text{куплено всего}$;

$\text{AveragePrecision@k} = \text{усредненный по сессиям пользователя Precision@k}$;

$\text{AverageRecall@k} = \text{усредненный по сессиям пользователя Recall@k}$;

Оценка качества. Online

Как проверить качество работы предложенной модели на практике?

- A/B тесты
- Оценка статистической значимости результата
- Продуктовые метрики

Оценка качества. Online

A/B тест

- Случайным образом делим пользователей на равные группы.
- Измеряем целевые метрики (например, количество заказов или доход) в каждой группе за длительный период времени.
- Получаем какое-то число для каждой группы.
- Пытаемся принять решение об успехе нашего подхода.

Оценка качества. Online

Статистическая значимость

- Приближение нормальным распределением
- Тест (критерий) Стьюдента
- Бутстреп

Оценка качества. Online

Продуктовые метрики

- Доход в группе
- Доход с пользовательской сессии
- Средняя стоимость купленного товара
- Средний чек
- Конверсия в покупку
- Клики
- Различные модели атрибуции: last click, first click

Заключение

- Ссылки:

- [Лекция К.В. Воронцов](#)
- [Мини курс на Coursera](#)
- [Рекомендательные системы: идеи, подходы, задачи](#)
- [Deep Learning based Recommender System: A Survey and New Perspectives](#)