Основы рекомендательных систем.

Повторение

Определение несбалансированности

- Данные **несбалансированы**, если число наблюдений одного класса сильно больше, чем число наблюдений других классов
- Что значит *сильно больше?*
- Явного порога нет, это зависит от задачи
- Соотношение классов 10:1 можно считать несбалансированностью

Accuracy: проблемы

- Алгоритму удобно предсказывать мажоритарный класс для всех наблюдений
- Мы должны изменить процедуру обучения и/или метрику качества

Точность и полнота

- **Точность (precision)** показывает, насколько сильно мы можем доверять нашему алгоритму, если он предсказывает положительный класс

- **Полнота (recall)** показывает долю наблюдений положительного класса, верно предсказываемых алгоритмом

F-мера: проблемы

- Точность, полнота и F-мера не учитывают True Negatives (TN) – количество верных предсказаний для наблюдений отрицательного класса

- Однако, если вас не интересуют True Negatives, это вполне нормально

Balanced accuracy

- True Positive Rate (полнота):

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

- True Negative Rate (специфичность):

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

Balanced accuracy

- **Balanced accuracy** – это среднее TPR and TNR

$$\text{Balanced accuracy} = \frac{\text{TPR} + \text{TNR}}{2}$$

MCC

- **Matthews correlation coefficient (MCC)** – это сбалансированная метрика, которая отражает корреляцию между правильными ответами и предсказаниями

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP})(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FP})(\mathrm{TN} + \mathrm{FN})}} \in [-1, 1]$$

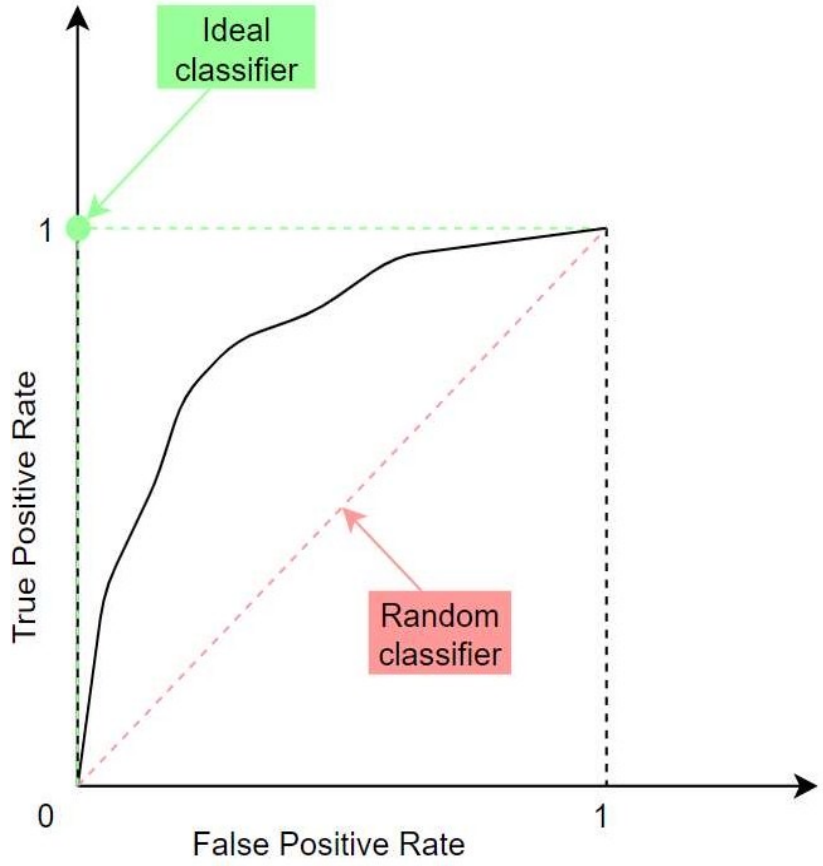
ROC-кривая и AUC-ROC

- При изменении t меняются значения TPR и FPR

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

- AUC-ROC – площадь под ROC-кривой



Веса классов

$$L(y, z) = -[y = 1] \times \log(z) - [y = -1] \times \log(1 - z)$$

- Штраф за ошибку в положительном наблюдении: $-\log(z)$
- Штраф за ошибку в отрицательном наблюдении: $-\log(1 - z)$

$$L(y, z) = -\mathbf{1000}[y = 1] \times \log(z) - [y = -1] \times \log(1 - z)$$

- Штраф за ошибку в положительном наблюдении: $-\mathbf{1000} \times \log(z)$
- Штраф за ошибку в отрицательном наблюдении: $-\log(1 - z)$

NearMiss

- Хотим контролировать процесс удаления объектов мажоритарного класса и сделать его менее случайным
- Будем использовать расстояния между объектами положительного и отрицательного классов
- Используем алгоритм kNN (k Nearest Neighbors) для определения близких и далеких объектов

Связи Томека

- Вместо сэмплирования напрямую, используем эвристики, которые позволят нам очистить данные

- Между объектами x и y разных классов существует **связь Томека**, если они являются ближайшими соседями друг друга:

$$\forall z: \qquad d(x,y) < d(x,z) \quad \text{and} \quad d(x,y) < d(y,z)$$

- z – другой объект
- $d(x,y)$ – расстояние между x и y

SMOTE

- **SMOTE: Synthetic Minority Over-sampling Technique**

- **Шаг 1.** Для каждого объекта миноритарного класса x_i найти k его ближайших соседей
- **Шаг 2.** Для каждого x_i выбрать среди его соседей M случайных: $x_i^{(1)}, \ldots, x_i^{(M)}$
- **Шаг 3.** Для каждой пары $(x_i, x_i^{(j)})$ сгенерировать новый объект:

$$x_i^{(j)\prime} = x_i + \lambda\left(x_i^{(j)} - x_i\right),$$

где $\lambda \in [0, 1]$ – случайное число.

ADASYN

- **ADASYN: ADAptive SYNthetic Sampling Approach**

- SMOTE:
  - **Шаг 2.** Для каждого объекта x_{i} сгенерировать M новых наблюдений

- ADASYN:
  - **Шаг 2.** Для каждого объекта x_{i} сгенерировать g_{i} новых наблюдений

Borderline-SMOTE

- Найти k ближайших соседей для каждого объекта x_i миноритарного класса
- Затем для каждого x_i вычислить $k' \in [0, k]$ – число соседей, принадлежащих к мажоритарному классу

1. Если $k' = k$, то x_i считаем шумом
2. Если $k' \in \left[0, \frac{k}{2}\right)$, то x_i – «надежный» объект (далеко от границы)
3. Если $k' \in \left[\frac{k}{2}, k\right)$, то x_i – объект «в опасности» (близко к границе)

Undersampling/oversampling

- В обоих методах модифицируется обучающая выборка – не валидация/тест!
- Разбиение на фолды для кросс-валидации нужно делать **до** oversampling
- Комбинация из undersampling и oversampling может неплохо сработать

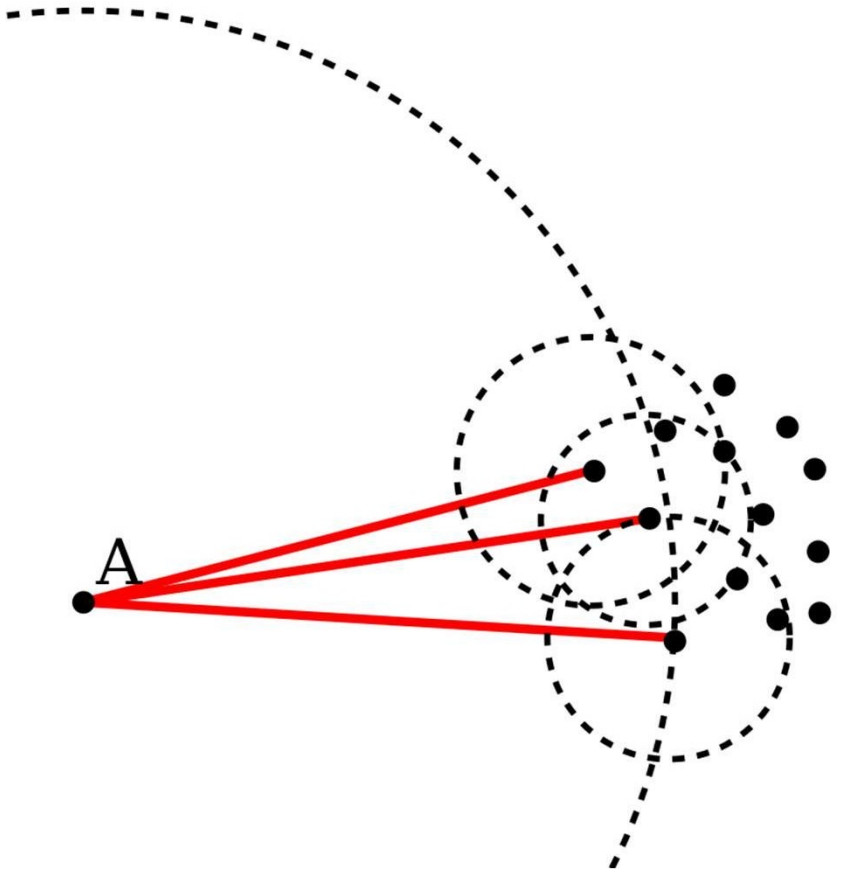
Методы на основе kNN

- Используем алгоритм kNN для детекции объектов, которые лежат далеко от остальных
- **Метод 1:** как далеко находится объект от своего k-ого ближайшего соседа
- **Метод 2:** какое среднее расстояние от объекта до k ближайших соседей?

Local Outlier Factor

- **LOF: Local Outlier Factor**

- Наблюдение аномально, если его локальная плотность намного меньше локальной плотности его ближайших соседей



Isolation Forest

- **Isolation Forest** «изолирует» наблюдения, делая случайные разбиения в решающих деревьях
- Идея: если наблюдение аномально, то чтобы его изолировать, нужно очень мало разбиений
- Построим лес и посчитаем оценку аномальности для каждого наблюдения

Рекомендательные системы

Рекомендательные системы

- Фильмы, видео
- Музыка
- Книги
- Приложения
- Товары
- Посты в социальных сетях
- Баннерные системы
- Люди (социальные сети, сервисы знакомств)
- Услуги (рестораны, отели, …)
- Научные публикации

Рекомендательные системы

- Рекомендательные системы сокращают объём информации, необходимый для принятия решения
- Не нужно читать отзывы на 1000 фильмов — модель сама выберет лучший

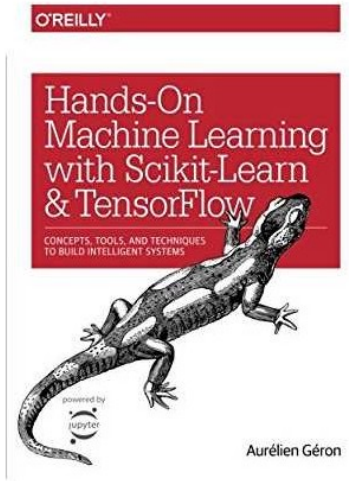
- Netflix: 2/3 просмотренных фильмов найдены через рекомендательную систему
- Amazon: 35% продаж через полки рекомендаций
- Youtube: 60% просмотров благодаря рекомендациям

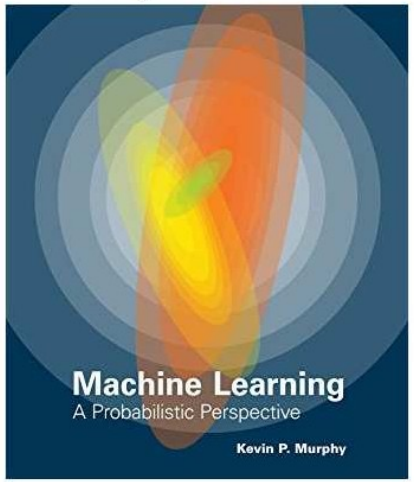
Amazon

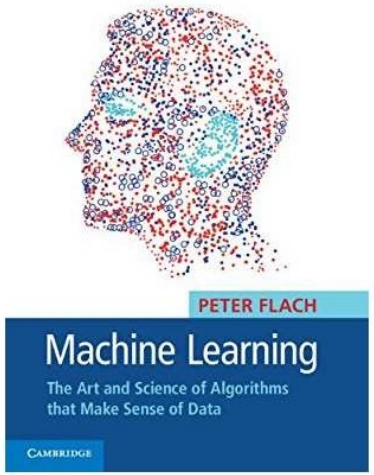
Try Amazon Prime today and get unlimited fast, FREE shipping See more

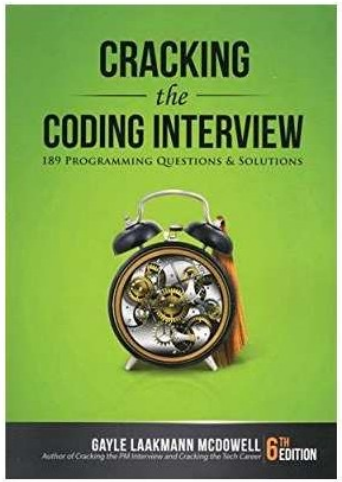
Amazon

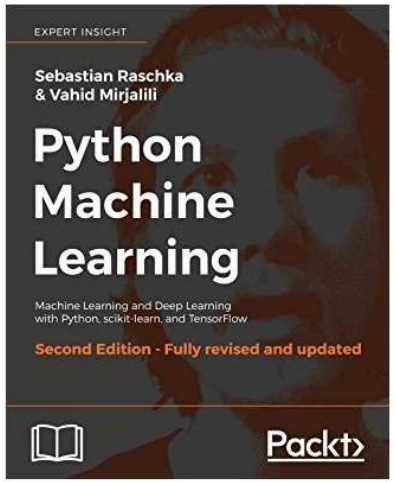
Recommendations for you in Books





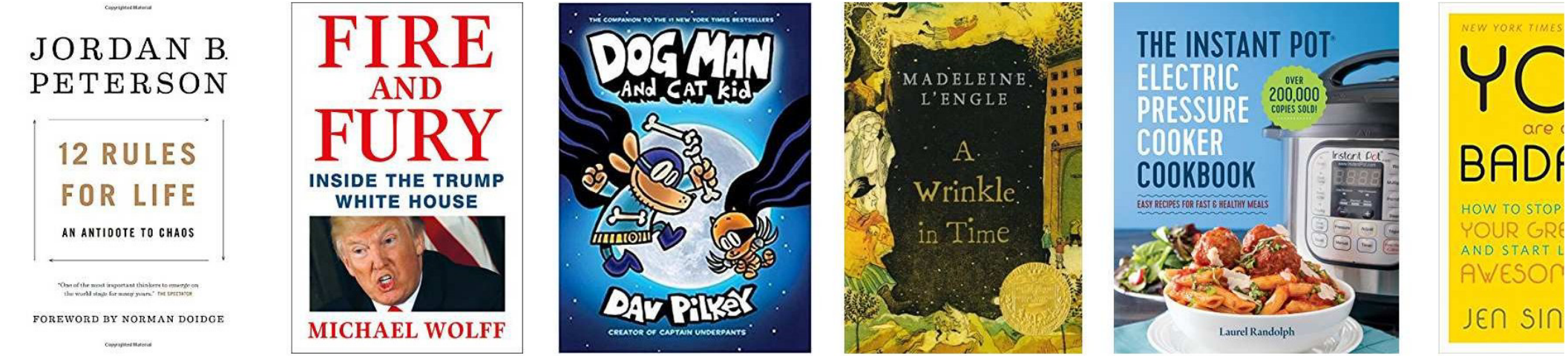






Amazon

Books best sellers See more



Netflix

Profile Type	Score Image A	Score Image B
Comedy	5.7	6.3
Romance	7.2	6.5

Image A

Image B

Рекомендации контента

- Медийный бум приводит к взрывному росту объёмов информации в сети
- Рекомендательные системы помогают ориентироваться
- Для авторов — поиск целевой аудитории

- Пионеры в Китае — Toutiao (более 100 миллионов активных пользователей) и другие платформы

Цели с точки зрения продавца

- Продать больше товаров
- Продать больше редких товаров
- Повысить лояльность пользователя
- Лучше понять покупателей

Цели с точки зрения покупателя

- Купить то, что нужно
- Понять, что покупать вместе с данным товаром
- Понять, что интересно (если нет задачи купить что-то конкретное)

Краткая история

- Начало 90-х: одна из первых рекомендательных систем (GroupLens, рекомендации записей в Usenet)
- Начало 2000-х: активные исследования, коммерциализация
- 2006: Netflix Prize
- 2007: первая конференция RecSys

Netflix Prize

- Предсказываем, какую оценку пользователь поставит фильму
- Метрика: RMSE
- Задача: улучшить на 10% качество предсказания
- Конкурс шёл с 02.10.2006 по 21.09.2009
- Главный приз: $1,000,000
- Размеры:
  - 500 тысяч пользователей
  - 17 тысяч фильмов
  - 8 рейтингов

Netflix Prize

- Одно из первых крупных соревнований по анализу данных (предшественник kaggle и т.д.)
- Первый большой открытый набор данных для тестирования алгоритмов рекомендаций
- Алгоритмы, разработанные участниками конкурса, до сих пор популярны в индустрии
- Netflix Prize привёл к большой популярности RMSE как метрики качества рекомендаций (не самый лучший результат)

Netflix Prize

На основе чего можно строить рекомендации?

- Данные по другим пользователям — «что смотрят люди с похожими на мои интересами?»
- Данные по объектам (фильмам) — «какие фильмы похожи на те, которые мне понравились?»

Типичная рекомендательная система

- Объект: пара «user-item»
- Целевая переменная: клики, длинные клики, досмотры, покупки, дослушивания, лайки и т.д.
- Решаем задачу классификации/регрессии/ранжирования

Особенности:

- Выбор целевой переменной
- Выбор метрики качества
- Факторы для модели
- Слишком много товаров/видео/песен/…

Отбор кандидатов

- Простая и быстрая модель, которая отбирает тысячи товаров для данного пользователя
- Сложная модель применяется только к отобранным кандидатам

Основные подходы

- Есть методы, разработанные напрямую для рекомендаций
- Коллаборативная фильтрация
  - Рекомендации на основе сходства действий пользователей
- Контентные рекомендации

Memory-based models

Обозначения

- Множество товаров:
- Множество пользователей:
- Множество пар «пользователь-товар», для которых известны оценки:
- Если для пары известен рейтинг, то будем писать
- Оценки — рейтинги фильмов, индикаторы покупки товара и т.д.

Оценки

- Оценки (или фидбэк) бывают явные и неявные
- Явные оценки
  - Пользователь поставил оценку фильму/товару
  - Пользователь написал отзыв
  - Пользователь поставил лайк
- Неявные оценки
  - Пользователь посмотрел фильм
  - Пользователь добавил товар в корзину
  - Пользователь долго смотрел на запись в социальной сети
- Неявные оценки более шумные, но их больше

Сходство пользователей

- $I_{uv} = \{i \in I \mid \exists r_{ui} \text{ и } \exists r_{vi}\}$ — множество товаров, которые оценили и пользователь u, и пользователь v
- Сходство пользователей (корреляция):

$$w_{uv} = \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \bar{r}_u)^2} \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \bar{r}_v)^2}},$$

где $\bar{r}_u$ и $\bar{r}_v$ — средние рейтинги пользователей

User-based collaborative filtering

- Дан пользователь u_0
- Найдём пользователей, которые похожи на него:

$$U(u_0) = \{v \in U \mid w_{u_0 v} > \alpha\}$$

- Порекомендуем те товары, которые часто покупались пользователями из $U(u_0)$

User-based collaborative filtering

Товары

Пользователи

1	1	0		1	
0	1	1			1
			1	1	0
	1	1		0	
	1				1

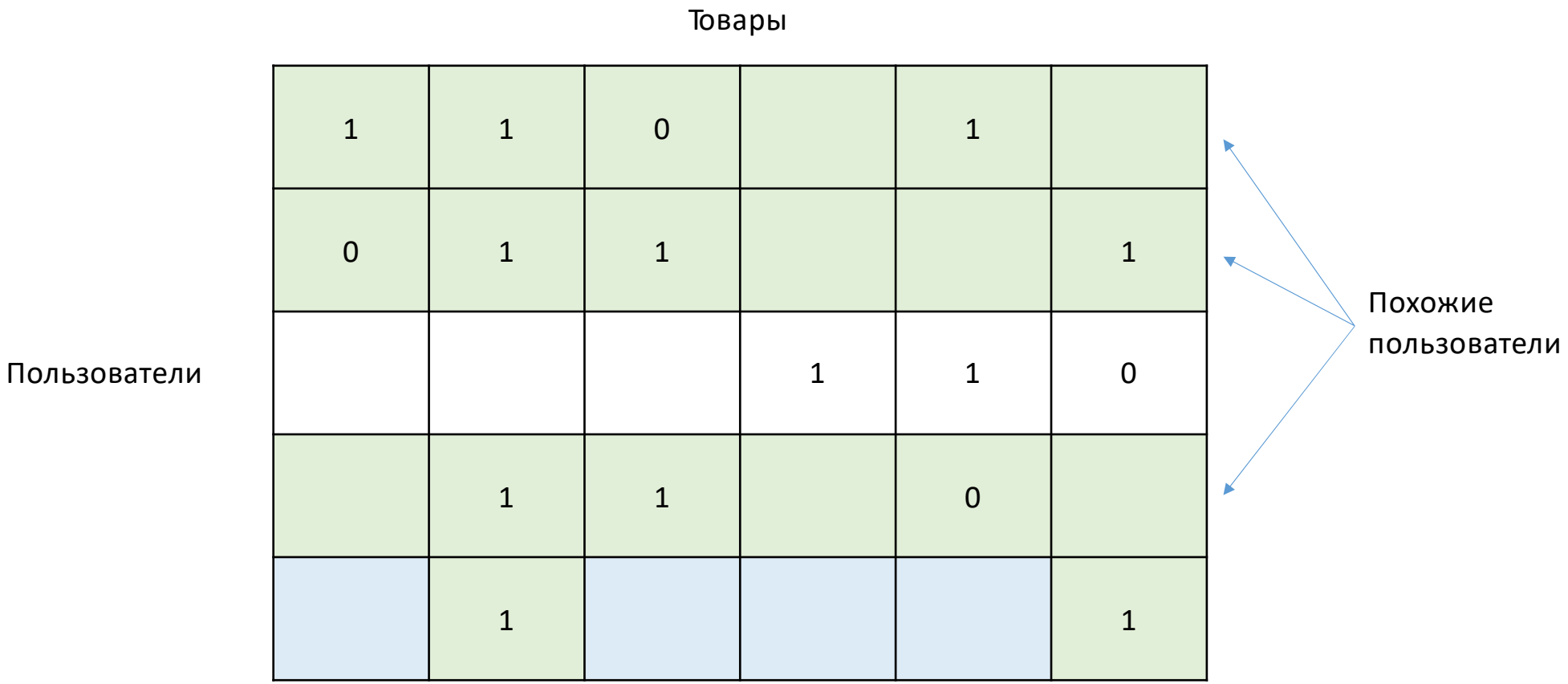
User-based collaborative filtering

Товары

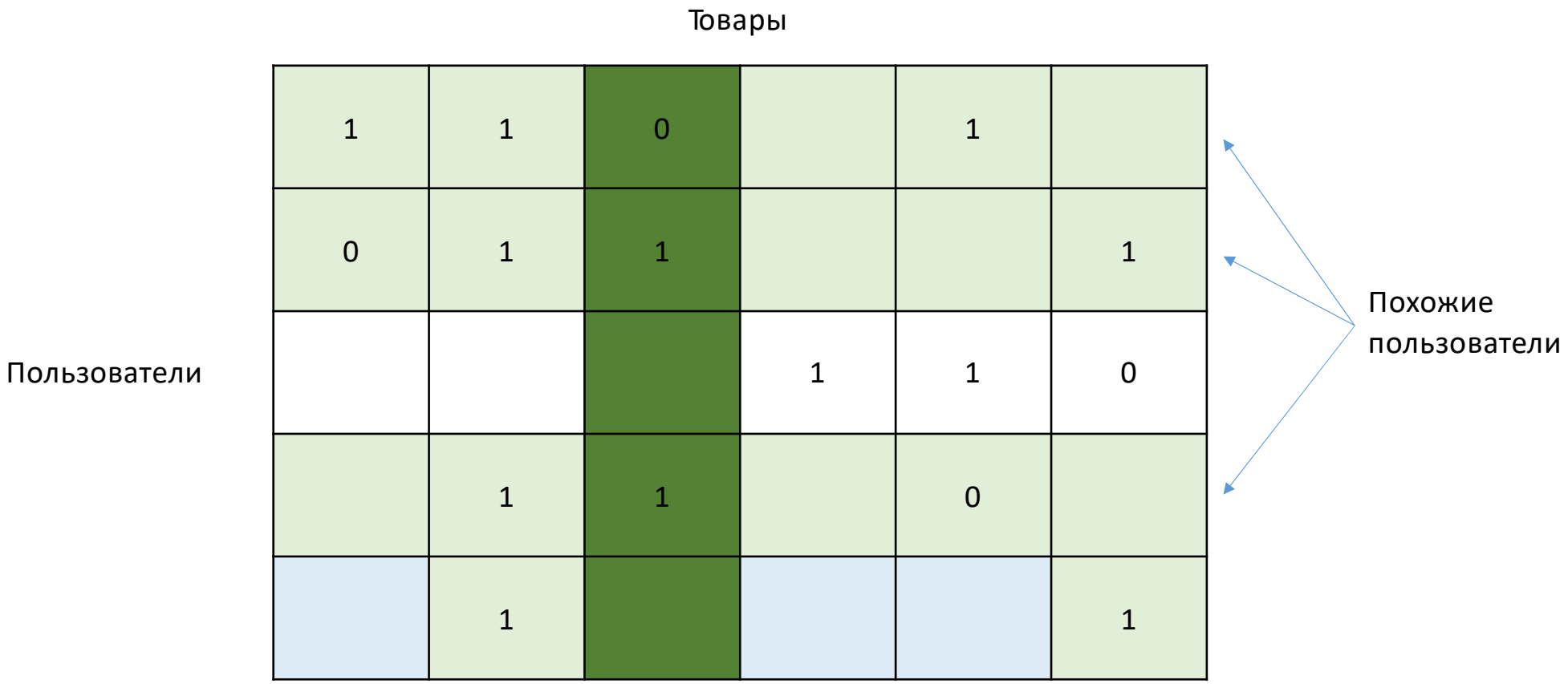
Пользователи

1	1	0		1	
0	1	1			1
			1	1	0
	1	1		0	
	1				1

User-based collaborative filtering



User-based collaborative filtering



User-based collaborative filtering

Недостатки:

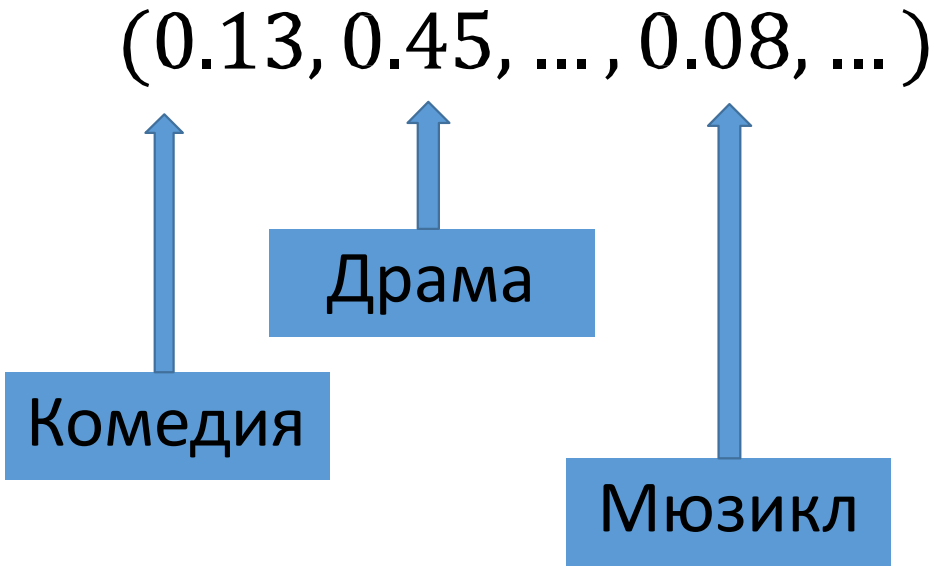
- Много параметров, которые сложно выбирать
  - Какой порог сходства для пользователей?
  - Сколько похожих пользователей должны были купить товар, чтобы мы его порекомендовали?
- Требуется хранить всю матрицу оценок

Есть и другие методы, основанные на сходствах, но все обладают теми же недостатками.

Модели со скрытыми переменными

Векторы интересов

- Для пользователя — насколько он интересуется каждым жанром
- Для фильма — насколько он относится к каждому жанру



Рейтинг

- Предположение: заинтересованность определяется как скалярное произведение векторов пользователя и фильма

$$(0.1, 0.5, 0.01, 0.92) \times (0, 0, 0.1, 0.95) = 0.875$$

$$(0.1, 0.5, 0.01, 0.92) \times (0.9, 0, 0, 0.1) = 0.182$$

Пользователь Фильм

Модели со скрытыми переменными

- Обучим вектор p_u для каждого пользователя u
- Обучим вектор q_i для каждого товара i
- Оценка приближается их скалярным произведением:

$$r_{ui} \approx \langle p_u, q_i \rangle$$

- Находим векторы только по известным оценкам
- После этого можем предсказать оценку для любой пары «пользователь-товар»

Модели со скрытыми переменными

- Оптимизационная задача:

$$\sum_{(u,i)\in R} \left(r_{ui} - \bar{r}_u - \bar{r}_i - \langle p_u, q_i \rangle\right)^2 \to \min_{P,Q}$$

- Решение: градиентный спуск, Alternating Least Squares (ALS) и другие методы

Модели со скрытыми переменными

	(0.9, 0.05)	(0.02, 1.1)	(1.05, 0.01)
(2.1, 5)	2	5	
(4.6, 0)	5		4
(0, 1)		1	
(4.9, 0.9)		1	5

Singular Value Decomposition (SVD)

Известно, что любая матрица размера $n \times k$ (ранга k) представима в виде $X = VDU^T$, где

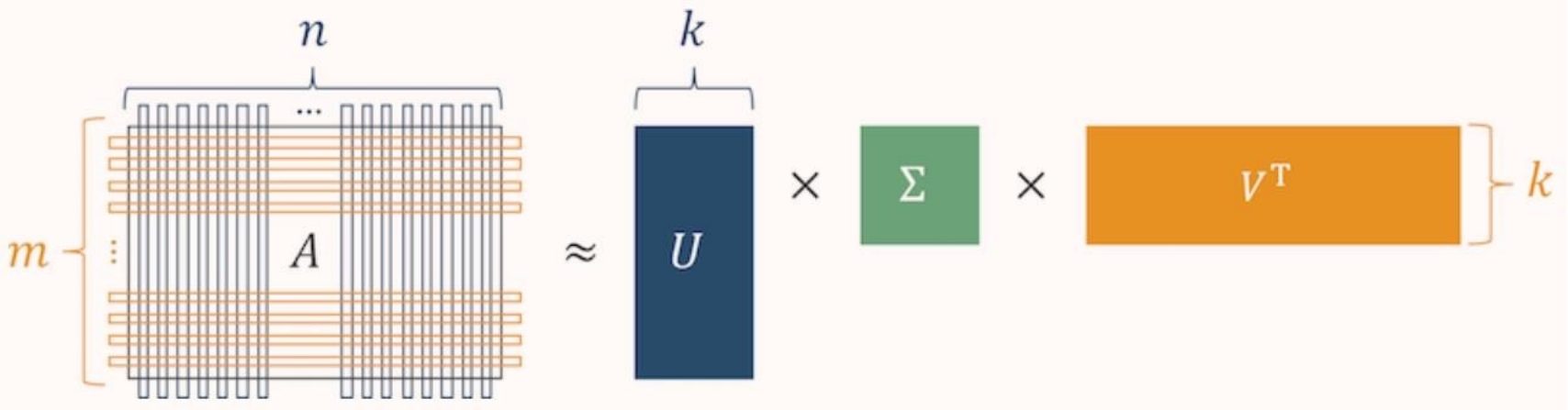
1) V – ортогональная матрица размера $n \times n$, ее столбцы – собственные векторы матрицы XX^T;

2) D – матрица размера $n \times k$, $d_{ii} = \sqrt{\lambda_i}$, $d_{ij} = 0$, если $i \neq j$, где $\{\lambda_i\}_{i=1}^{k}$ – собственные числа матрицы X^TX (и ненулевые собственные значения матрицы XX^T);

3) U – ортогональная матрица размера $k \times k$, её столбцы – собственные векторы матрицы X^TX.

SVD для построения рекомендаций

- Матрица товарных предпочтений (матрица, где строки это пользователи, а столбцы это продукты, с которыми пользователи взаимодействовали) представляется произведением трех матриц:

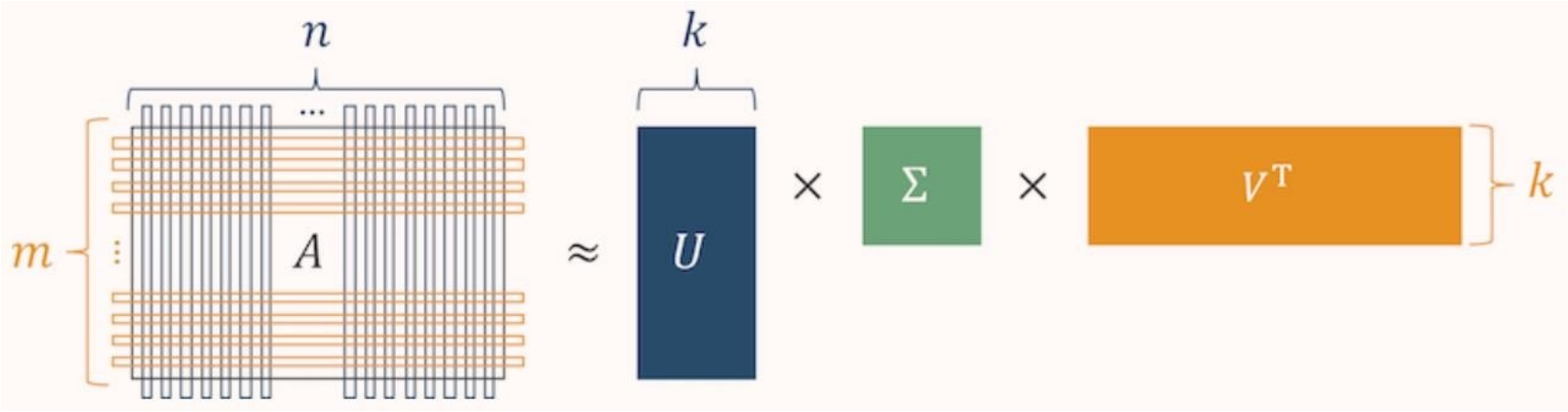


U – описание характеристик пользователя

V – описание характеристик продукта

SVD для построения рекомендаций

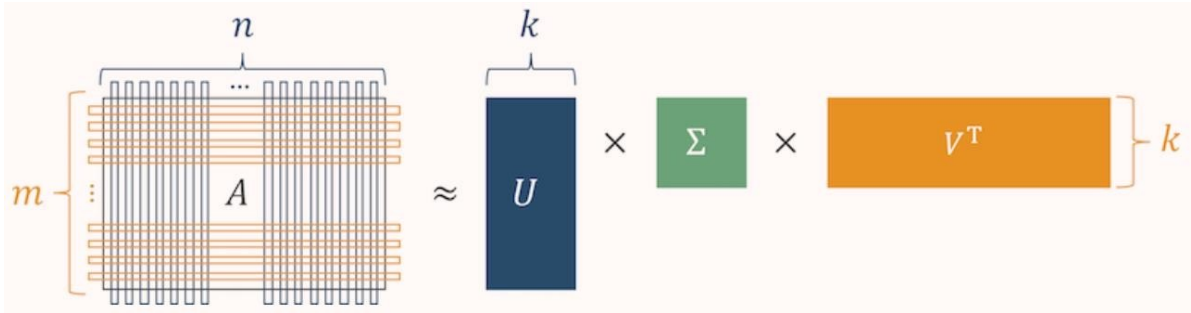
- Матрица товарных предпочтений (матрица, где строки это пользователи, а столбцы это продукты, с которыми пользователи взаимодействовали) представляется произведением трех матриц:



- После восстановления исходной матрицы, клетки, где у пользователя были нули, а появились «большие» числа, показывают степень латентного интереса к товару. Упорядочим эти цифры, и получим список товаров, релевантных для пользователя.

SVD для построения рекомендаций

- Матрица товарных предпочтений (матрица, где строки это пользователи, а столбцы это продукты, с которыми пользователи взаимодействовали) представляется произведением трех матриц:

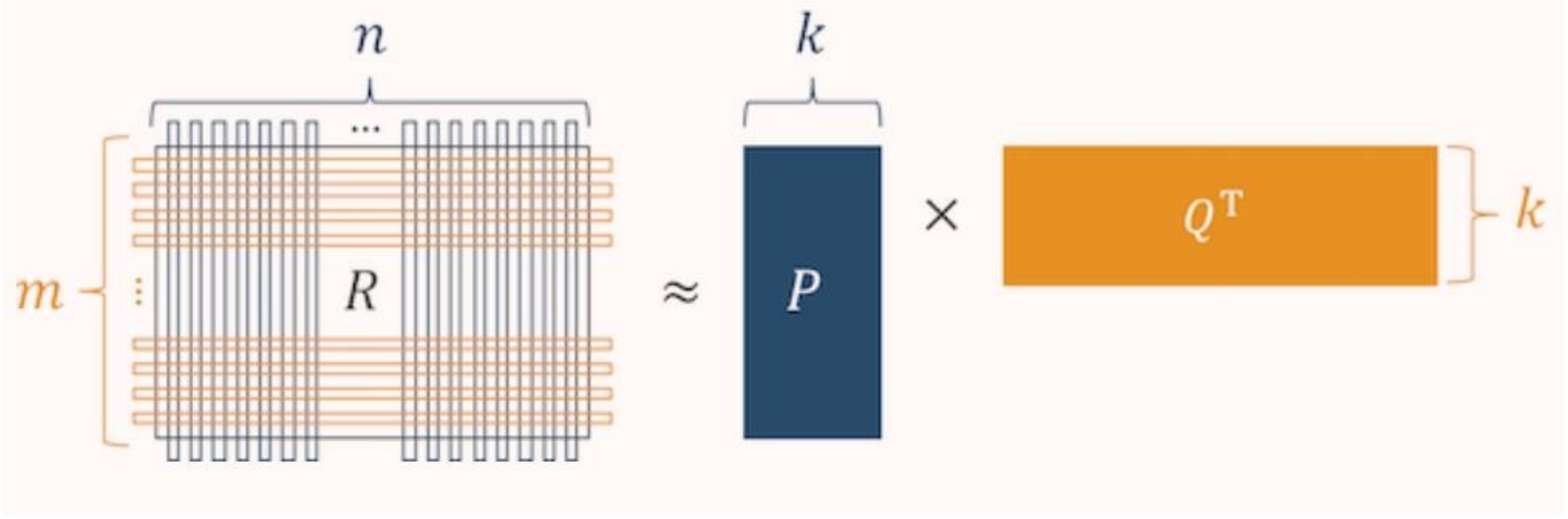


- После восстановления исходной матрицы, клетки, где у пользователя были нули, а появились «большие» числа, показывают степень латентного интереса к товару. Упорядочим эти цифры, и получим список товаров, релевантных для пользователя.
- При этой операции у пользователя и товара появляются «латентные» признаки. Это признаки, показывающие «скрытое» состояние пользователя и товара.

Общее семейство подобных алгоритмов называется *NMF (non- negative matrix factorization)*. Как правило вычисление таких разложений весьма трудоемко, поэтому на практике часто прибегают к их приближенным итеративным вариантам.

Матричная факторизация (факторизационные машины) с помощью ALS

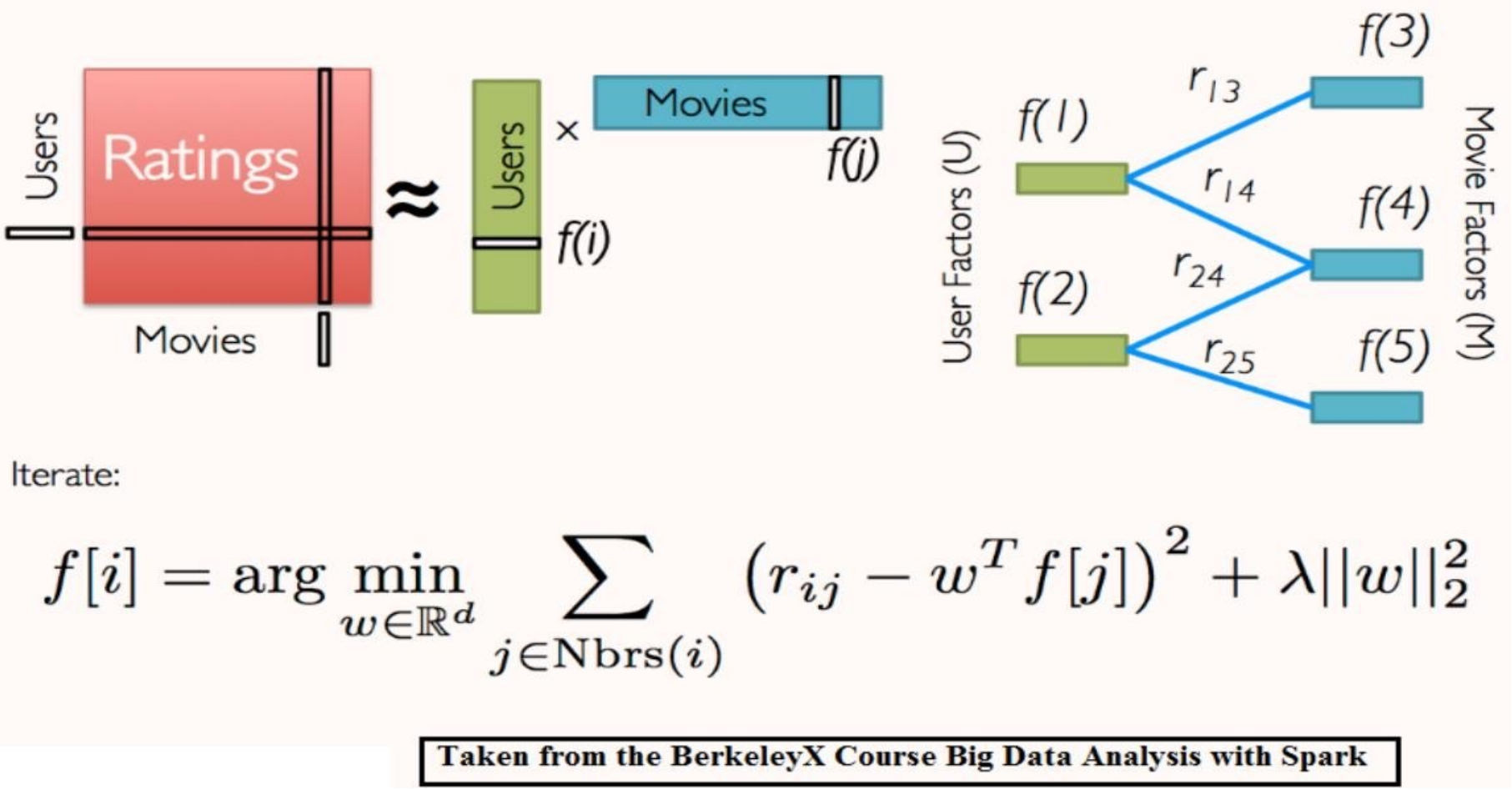
- **ALS (alternating least squares):** популярный итеративный алгоритм разложения матрицы предпочтений на произведение 2 матриц: факторов пользователей (U) и факторов товаров (Q)



Матричная факторизация с помощью ALS

- **ALS (alternating least squares):** популярный *итеративный алгоритм разложения матрицы предпочтений на произведение 2 матриц: факторов пользователей (U) и факторов товаров (I)*.
- **Принцип работы:** *минимизация среднеквадратичной ошибки на проставленных рейтингах*.
- **Оптимизация** происходит поочередно, сначала по факторам пользователей, потом по факторам товаров.
- **Для обхода переобучения** к среднеквадратичной ошибке добавляются регуляризационные коэффиценты.

Матричная факторизация



Iterate:

$$f[i] = \arg\min_{w \in \mathbb{R}^d} \sum_{j \in \mathrm{Nbrs}(i)} \left(r_{ij} - w^T f[j]\right)^2 + \lambda ||w||_2^2$$

Taken from the BerkeleyX Course Big Data Analysis with Spark

https://habr.com/ru/company/lanit/blog/421401/

Матричная факторизация

- Мы можем дополнить матрицу предпочтений новым измерением, содержащим информацию о пользователе или товаре. Таким образом, мы задействуем больше доступной информации и возможно получим более точную модель.

На практике именно факторизационные машины в большинстве кейсов дают наилучший результат!

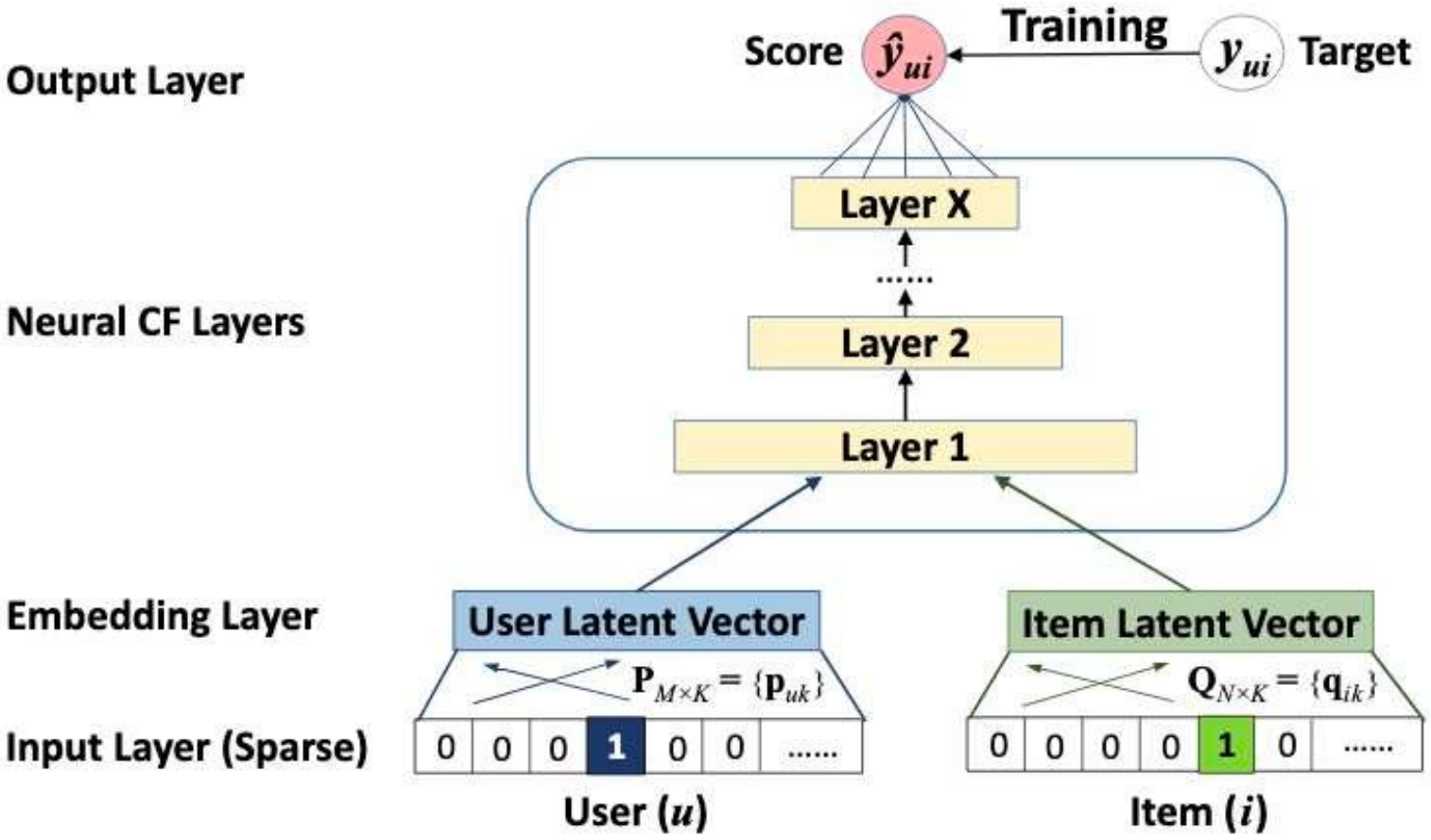
Контентные методы

Контентные рекомендации

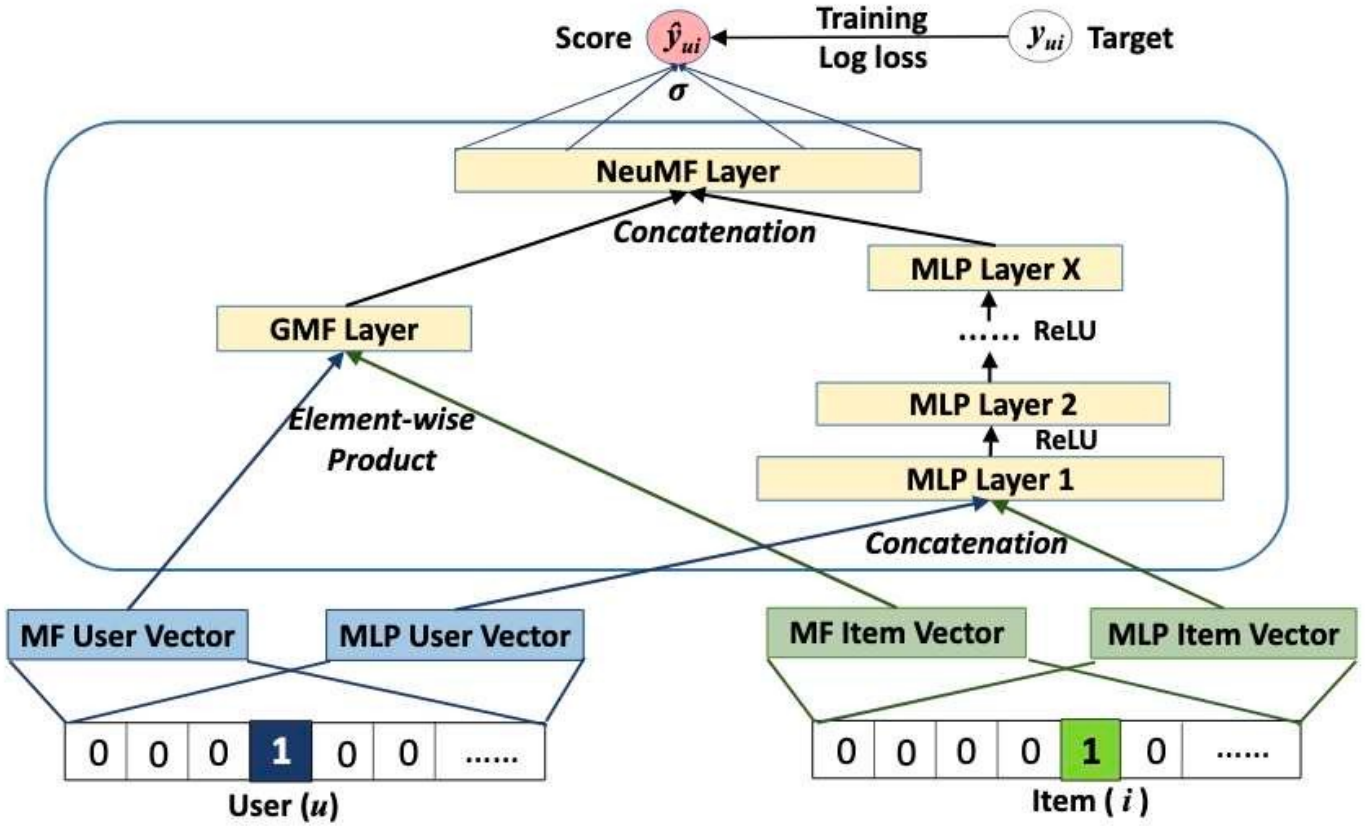
- Сведём задачу к обычному обучению с учителем
- Объект: пара «пользователь-товар»
- Ответ: отклик пользователя
- Факторы: информация про пользователя и про товар
- Обучаем любую модель на этих данных
- Среди факторов могут быть и прогнозы коллаборативных моделей

Нейросетевые методы

Neural Collaborative Filtering



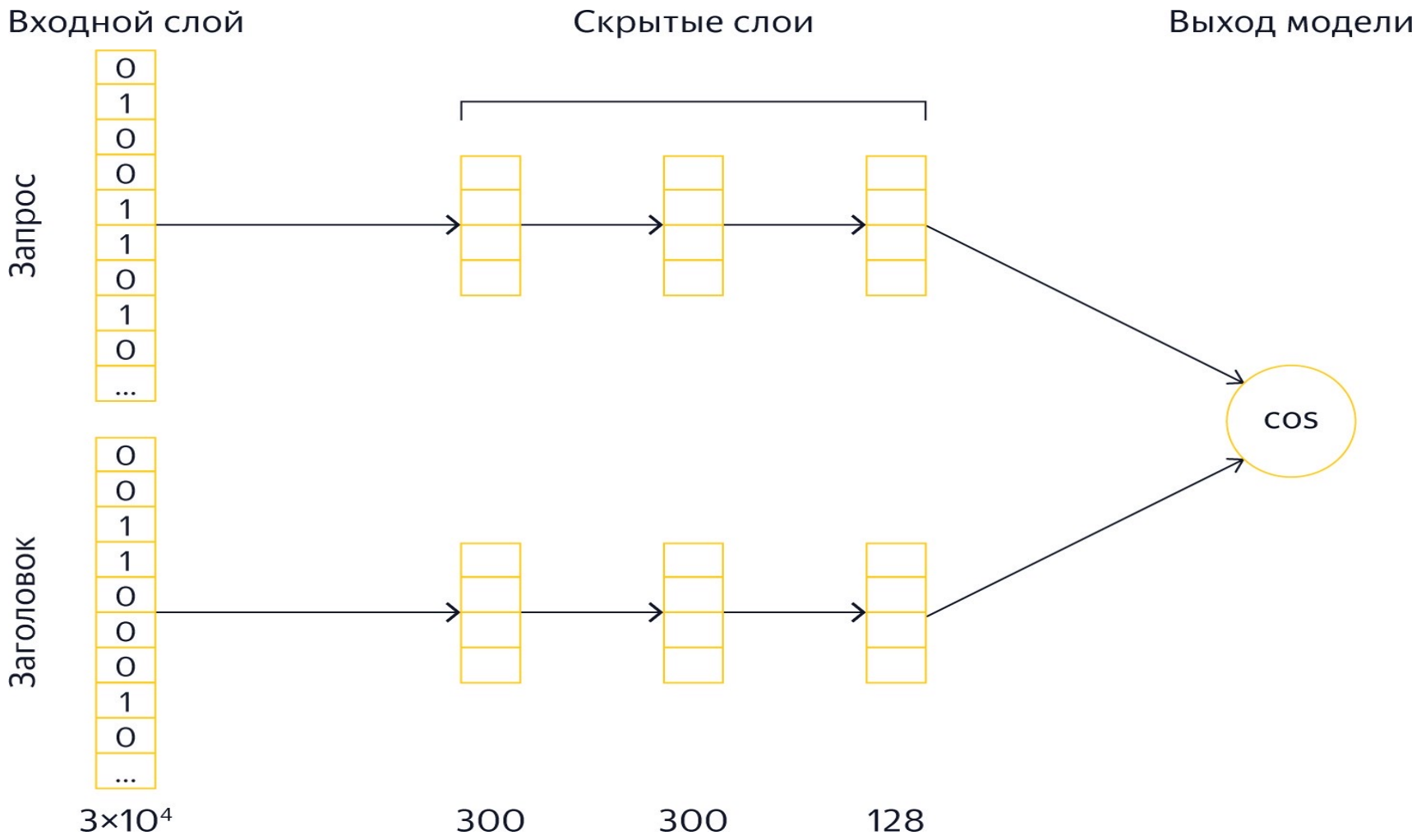
Neural Collaborative Filtering



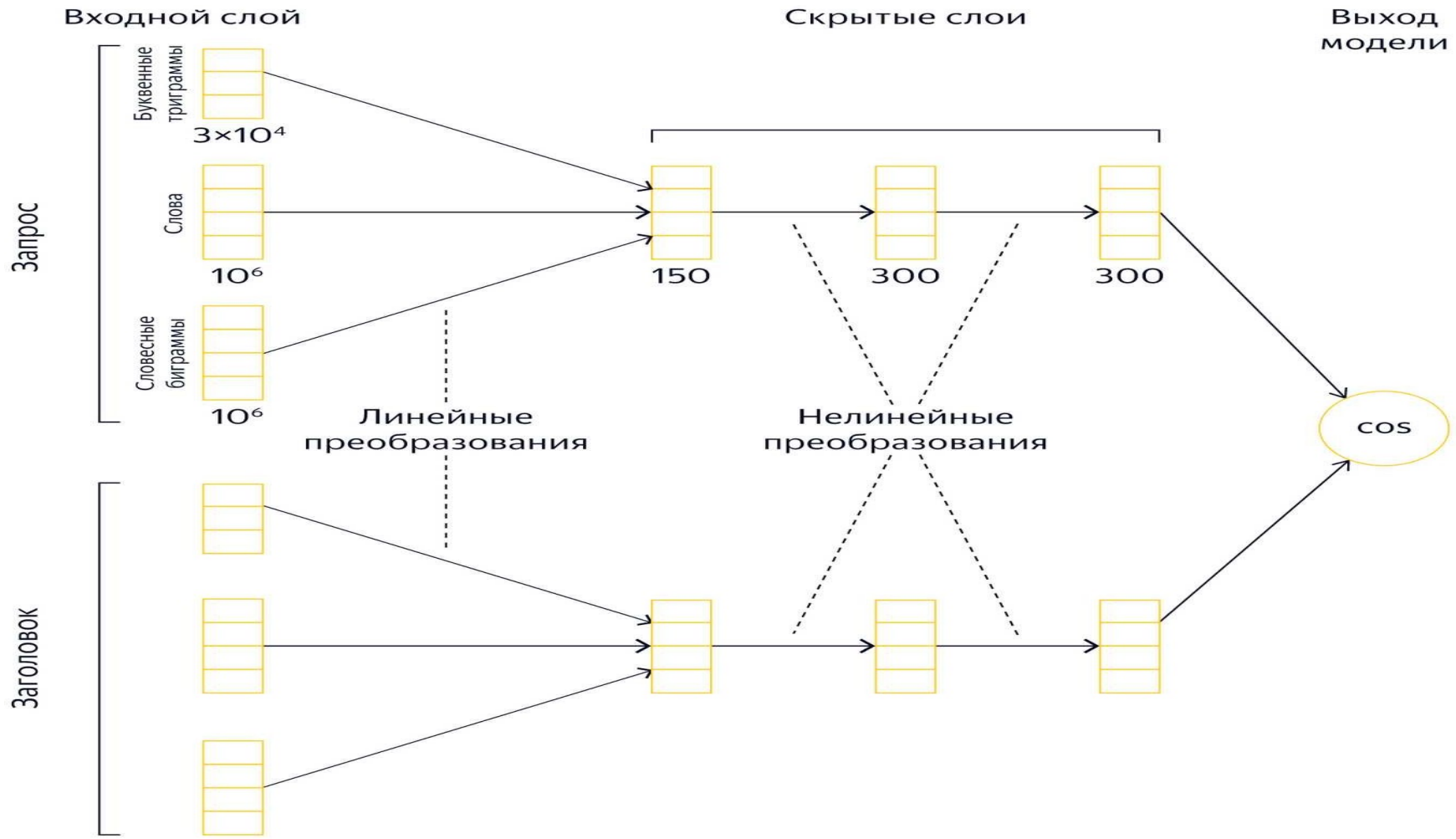
Word-hashing

- К тексту добавляются маркеры начала и конца
- После чего он разбивается на буквенные триграммы
- Пример: [палех] –> [па, але, лех, ех]

Deep Structured Semantic Model



Deep Structured Semantic Model



Метрики качества рекомендаций

Качество предсказаний

В зависимости от целевой переменной:

- MSE, MAE, R^2
- Accuracy, HitRate, precision/recall, AUC-ROC
- Метрики качества ранжирования (дальше в курсе)

Качество предсказаний

- Насколько хорошо мы предсказываем оценки ?
- Разделяем сессии пользователей на две части: обучаемся на первой, измеряем качество предсказания на второй
- Оцениваем, насколько хорошо предсказываем поведение пользователя — но не факт, что нужно именно это
- Зачем рекомендовать то, что он и так купил бы?

Другие метрики

- Покрытие
  - Какая доля товаров рекомендовалась хотя бы раз?
  - Какой доле пользователей хотя бы раз показаны рекомендации?
- Новизна
  - Как много рекомендованных товаров пользователь встречал раньше?
- Прозорливость (serendipity)
  - Способность предлагать товары, которые отличаются от купленных ранее
- Разнообразие

Резюме

- Рекомендации — широкая задача с большим количеством коммерческих применений
- Модели: коллаборативная фильтрация, контентный подход
- Рекомендации товаров на основе сходства пользователей
- Модели со скрытыми переменными
- Обилие метрик качества

Почитать

- https://www.benfrederickson.com/approximate-nearest-neighbours-for-recommender-systems/
- https://habr.com/ru/company/yandex/blog/314222/
- https://www.jefkine.com/recsys/2017/03/27/factorization-machines/
- https://www.csie.ntu.edu.tw/~b97053/paper/Rendle2010FM.pdf
- https://arxiv.org/abs/1708.05031

Спасибо за внимание!

Ildar Safilo

@Ildar_Saf
irsafilo@gmail.com
https://www.linkedin.com/in/isafilo/