

# Промышленное машинное обучение на Spark

Лекция 7: Рекомендательные  
системы

01

Оборачиваем модель в сервис. Первая часть: Docker. Flask

02

Оборачиваем модель в сервис. Вторая часть: Requests. REST API

03

Распределенные вычисления. HDFS. MapReduce. Spark DataFrame

04

Погружение в среду Spark. RDD, SQL, Pandas API.

05

Генерация признаков. Spark feature engineering

06

Распределенное обучение моделей. Spark ML

07

Рекомендательные системы. Spark NLP

08

Обработка потоковых данных. Spark Streaming.

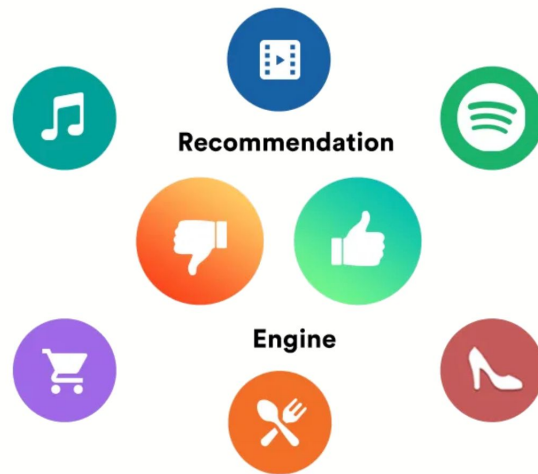
# Рекомендательные системы. Метрики. Виды

## План:

1. Рекомендательные системы и их задачи
2. Content-based & Collaborative filtering
3. Implicit & Explicit feedback
4. Метрики рекомендательных систем
5. Матричные разложения. ALS.
6. Реализации в Spark ML

# Задачи рекомендательных систем

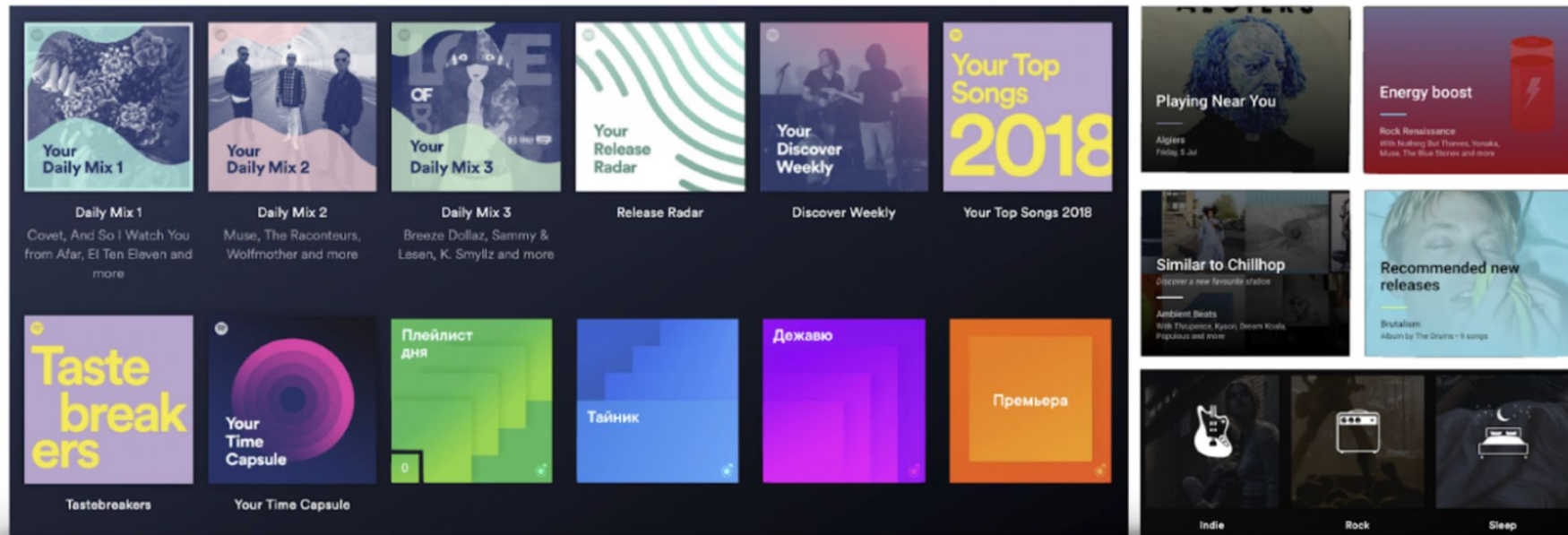
1. Предмет рекомендации - что рекомендуем?
2. Цель рекомендации - зачем рекомендуем?
3. Контекст рекомендации - что пользователь делает в этот момент?
4. Источник рекомендации - кто рекомендует?



# Аспекты рекомендательных систем

1. Уровень персонализации - насколько точно система может строить портрет пользователя?
2. Уровень гибкости - Как быстро умеет адаптироваться к новым информации и источникам?
3. Прозрачность рекомендации - насколько результаты рекомендации интерпретируемые?
4. Объём базы знаний - как много информации было задействовано при рекомендации?

# Классический пример - рекомендация музыки



Нравится **инди**? А это слышали? Если вам нравится **Сатана Печёт Блины**, обратите внимание на эту музыку

**Стоунер-рок**: популярное за неделю

Это слушают люди, которым нравится то же, что и вам

**Cigarettes After Sex**: что в плейлистах у поклонников

**Ваш друг** слушает — и вы послушайте

# Виды рекомендательных систем

Существует огромное множество различных алгоритмов рекомендательных систем, но все они делятся на три основных вида:

## 1. Рекомендации на основе контента (Content-based)

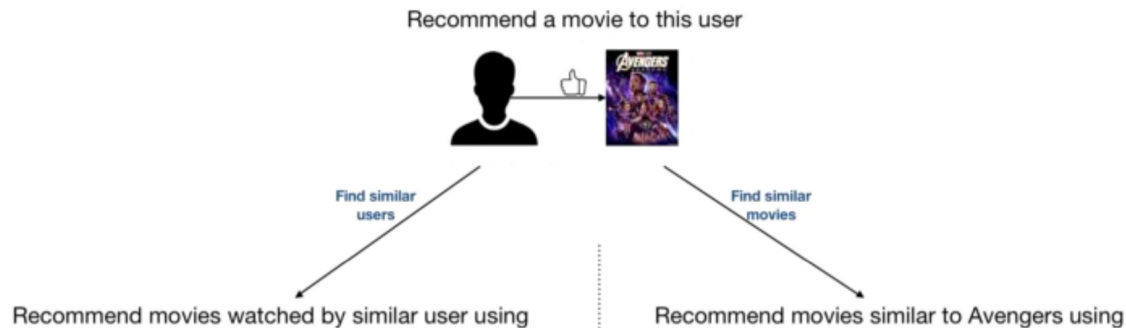
- Рекомендации строятся на основе объектов, которые похожи на те объекты, с которыми пользователь взаимодействовал
- Уровень подобия оценивается только по признакам самого объекта
- Можем рекомендовать объекты, которые находятся в одной предметной области

## 2. Коллаборативная фильтрация (Collaborative Filtering):

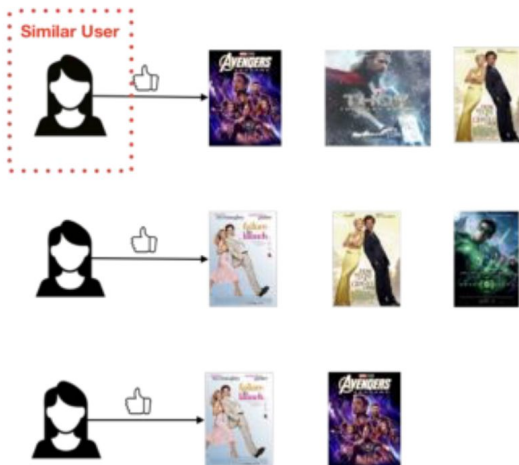
- Следующие рекомендации строятся на основе пользовательской истории взаимодействия и истории взаимодействия других пользователей
- Нет привязки к предметной области
- Не умеет работать с новыми пользователями

## 3. Гибридные подходы - учитывают при рекомендациях внутренние особенности объекта, так и историю взаимодействия с другими объектами

# Content-based & Collaborative filtering



## Collaborative Filtering



## Content Based Filtering





# Классическая постановка задачи рекомендации

В классической постановке задачи всё, что у нас есть — это матрица оценок user-item. Она очень разрежена и наша задача — заполнить пропущенные значения. Обычно в качестве метрики используют RMSE предсказанного рейтинга, но есть мнение, что это не совсем правильно и следует учитывать характеристики рекомендации как целого, а не точность предсказания конкретного числа.

	M1	M2	M3	M4	M5
	3	1	1	3	1
	1	2	4	1	3
	3	1	1	3	1
	4	3	5	4	4

# Виды пользовательских фидбеков

Чтобы построить рекомендательную систему, необходимо учитывать пользовательские отклики на тот или иной объект рекомендации.

- **Явный отклик (Explicit feedback)** - когда можем собрать прямую оценку/мнение пользователя об объекте, например, в случае с рекомендацией фильмов - это может быть поставленный пользователем рейтинг данному фильму
- **Неявный отклик (Implicit feedback)** - более часто встречаемый вид откликов, когда нет прямой оценки от пользователя об объекте, поэтому оценка аппроксимируется исходя совершенных действий: клики, просмотры, добавление в корзину и тд.

# Метрики рекомендательных систем

При построение любой системы важно уметь правильно оценивать качество предлагаемого решения. В случае с задачей рекомендации существует два подхода в оценке алгоритмов.

- **Online-evaluation** - самый точный способ оценки качества системы — прямая проверка на пользователях в контексте бизнес-метрик. Это может быть CTR, время, проведенное в системе, или количество покупок. Но эксперименты на пользователях дороги, а выкатывать плохой алгоритм даже на малую группу пользователей не хочется, поэтому до онлайн-проверки пользуются оффлайн метриками качества.
- **Offline-evaluation** - оценка качества производится на отложенной выборке, которая как правило собирается из истории взаимодействия пользователей с системой. Как правило, для оценки применяют те же метрики, что и для задач поиска и ранжирования: MAP@K, nDCG@K и тд.

# Offline метрики

Также помимо метрик ранжирования, если имеется точное значение оценки, применяется метрика RMSE:



$$RMSE = \sqrt{\frac{1}{|\mathcal{D}|} \sum_{(u,i) \in \mathcal{D}} (\hat{r}_{ui} - r_{ui})^2}$$

[Статья с хабра с подробным объяснением метрик ранжирования](#)

Решение задачи рекомендации через матричные разложения. Алгоритм ALS.

# Матрица взаимодействий

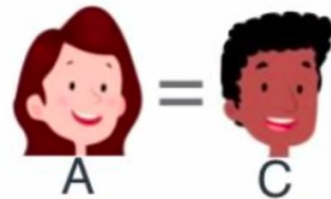
Так как матрица взаимодействий - это математический объект, то к нему применимы математические операции разложения, а также поиск взаимозависимых строк и столбцов

	M1	M2	M3	M4	M5
	3	1	1	3	1
	1	2	4	1	3
	3	1	1	3	1
	4	3	5	4	4

# Зависимости между строками

	M1	M2	M3	M4	M5
	3	1	1	3	1
					
	3	1	1	3	1
					

Two large red arrows point from the right towards the first and third rows of the table, indicating a dependency between them.



# Зависимости между столбцами





	M1	M2	M3	M4	M5
	3			3	
	1			1	
	3			3	
	4			4	



**M1 = M4**



# Линейные комбинации предпочтений





	M1	M2	M3	M4	M5
					
	1	2	4	1	3
	3	1	1	3	1
	4	3	5	4	4

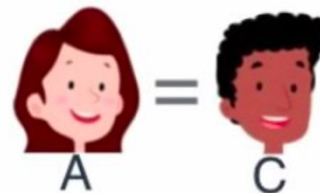
Three red arrows point to the last three rows of the table.

$$\begin{array}{c} \text{Woman with brown hair} \\ B \end{array} + \begin{array}{c} \text{Man with dark skin and hair} \\ C \end{array} = \begin{array}{c} \text{Woman with blonde hair} \\ D \end{array}$$

# Предсказания на основе соотношений

Если известно, какие соотношения между строками или столбцами, то можно делать прогнозы незаполненных значений на основе имеющейся информации

	M1	M2	M3	M4	M5
	3	1	1	3	1
	1	2	4	1	3
	3	1	1	3	
	4	3	5	4	4



Movie 5

# Какие факторы формируют пользовательскую оценку



Has a Sad Dog



Comedy



Drama



Action



Meryl Streep



Sexy Canadian Ryan



Big Boat



Scary



# Математический допущения алгоритма


Предположим, что существует некоторое признаковое пространство  $U$  и  $I$  размерности  $d$ , как для пользователей, так и для фильмов.

Теперь каждый пользователь описывается некоторым вектором признаков:  $u = (u_1, \dots, u_d)$  и каждый предмет описывается своим вектором  $i = (i_1, \dots, i_d)$

Тогда будем предсказывать оценку  $r_{u,i}$  некоторой предмета  $i$  пользователем  $u$  по следующей формуле  $r_{u,i} = u * i^T$

# Матричное разложение



	 Comedy	 Action
M1	3	1
M2	1	2
M3	1	4
M4	3	1
M5	1	3















	 Comedy	 Action
		
		
		
		





=

	M1	M2	M3	M4	M5
	3	1	1	3	1
	1	2	4	1	3
	3	1	1	3	1
	4	3	5	4	4

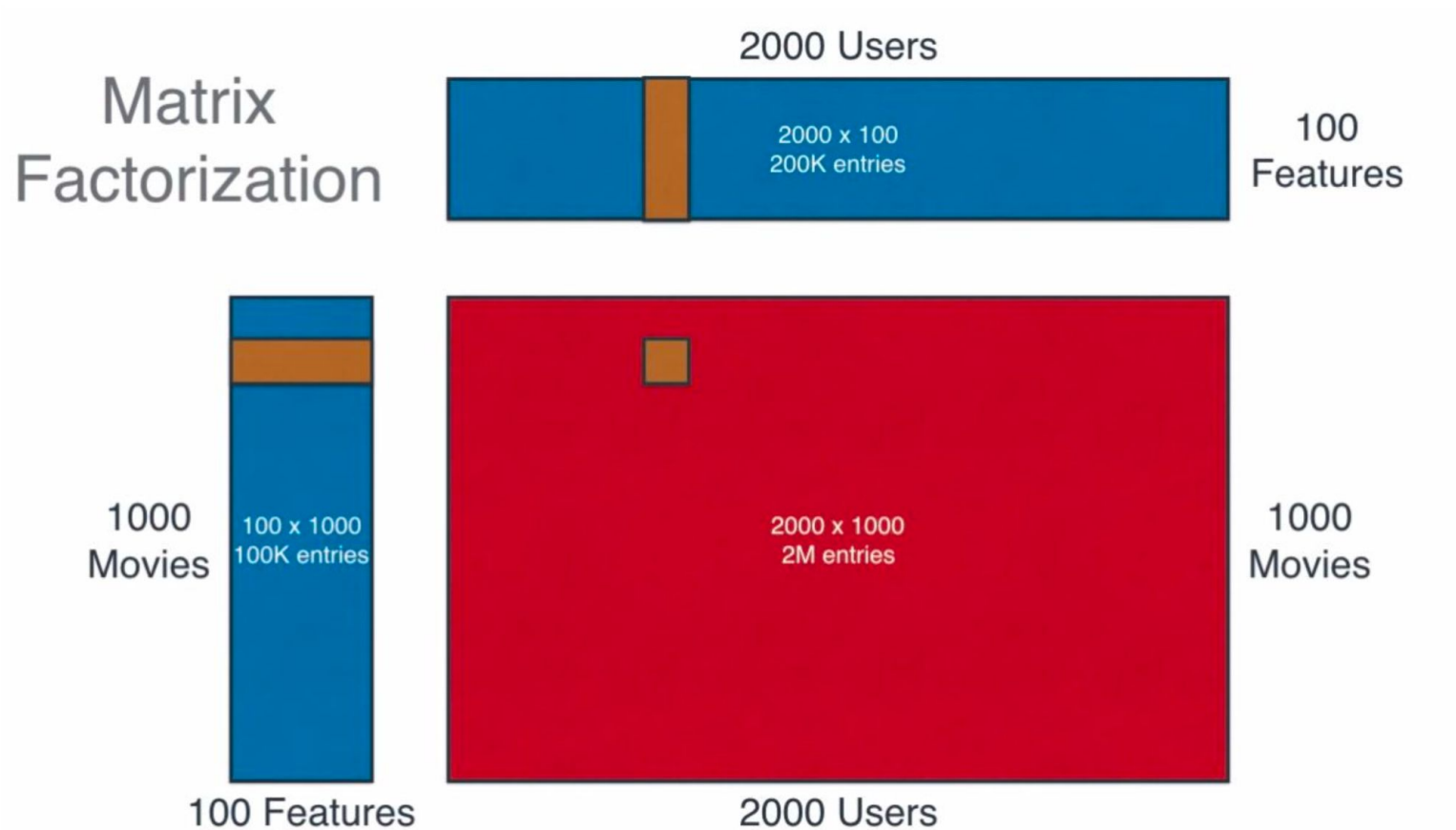
# Результат матричного произведения

	M1	M2	M3	M4	M5
 Comedy	3	1	1	3	1
 Action	1	2	4	1	3

	 Comedy	 Action
 A		
 B		
 C		
 D		

	M1	M2	M3	M4	M5
	3	1	1	3	1
	1	2	4	1	3
	3	1	1	3	1
	4	3	5	4	4

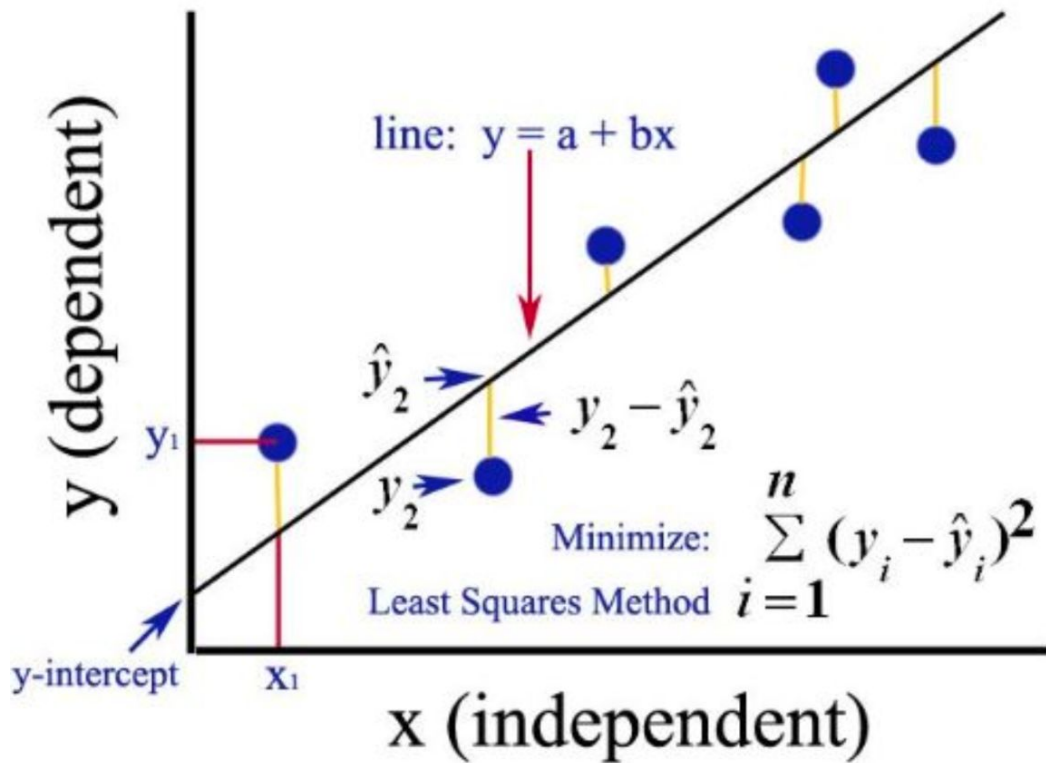
# Целевая матрица



Как получить нужные нужные разложения?



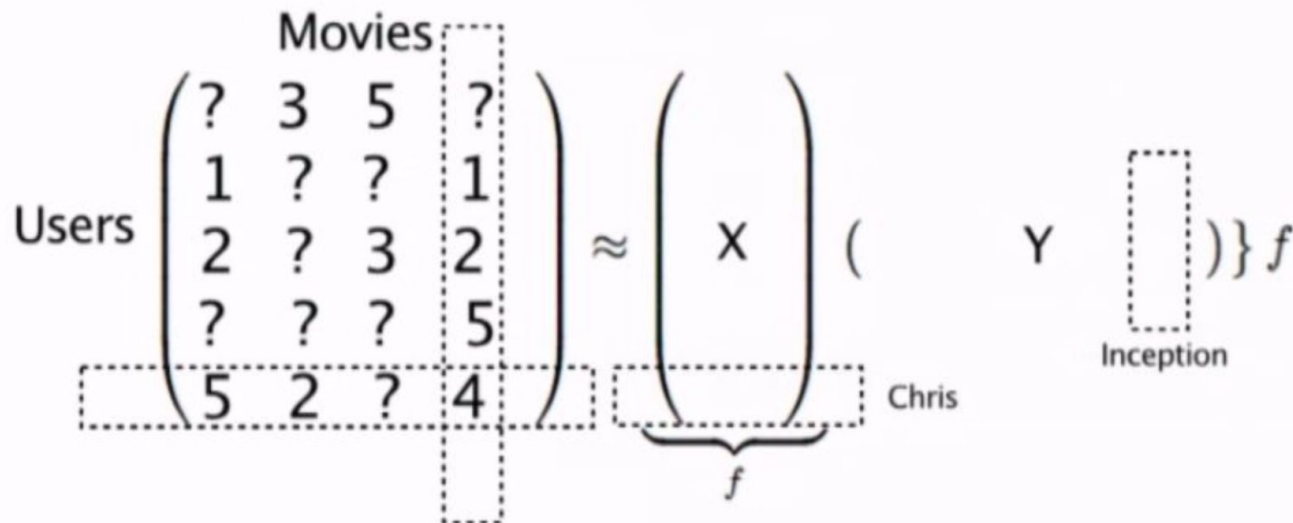
# Метод наименьших квадратов в линейной регрессии



Метрика, которую оптимизируем

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (p_i - y_i)^2}{n}}$$

Теперь та же самая задача только для матриц



$$\min_{x,y} \sum_{u,i} (r_{ui} - x_u^T y_i - \beta_u - \beta_i)^2 + \lambda (\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2)$$

- $r_{ui}$  = user  $u$ 's rating for movie  $i$
- $x_u$  = user  $u$ 's latent factor vector
- $x_i$  = item  $i$ 's latent factor vector
- $\beta_u$  = bias for user  $u$
- $\beta_i$  = bias for item  $i$
- $\lambda$  = regularization parameter

# Ищем матрицы U и I методом ALS

Будем искать матрицы U и I итеративным алгоритмом - поочерёдно фиксируя матрицу пользователей и матрицу предметов

Таким образом задача превращается в последовательность задач наименьших квадратов

$$\min_{x,y} \sum_{u,i} (r_{ui} - x_u^T y_i - \beta_u - \beta_i)^2 + \lambda (\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2)$$

- $r_{ui}$  = user  $u$ 's rating for movie  $i$
- $x_u$  = user  $u$ 's latent factor vector
- $x_i$  = item  $i$ 's latent factor vector
- $\beta_u$  = bias for user  $u$
- $\beta_i$  = bias for item  $i$
- $\lambda$  = regularization parameter

# Сам алгоритм

---

## SGD Algorithm for MF

---

**Input:** training matrix  $V$ , the number of features  $K$ , regularization parameter  $\lambda$ , learning rate  $\epsilon$

**Output:** row related model matrix  $W$  and column related model matrix  $H$

- 1: Initialize  $W, H$  to  $UniformReal(0, \frac{1}{\sqrt{K}})$
  - 2: **repeat**
  - 3:   **for random**  $V_{ij} \in V$  **do**
  - 4:      $error = W_{i*}H_{*j} - V_{ij}$
  - 5:      $W_{i*} = W_{i*} - \epsilon(error \cdot H_{*j}^\top + \lambda W_{i*})$
  - 6:      $H_{*j} = H_{*j} - \epsilon(error \cdot W_{i*}^\top + \lambda H_{*j})$
  - 7:   **end for**
  - 8: **until** convergence
-

# Как подобрать число $d$ - размерность признаков?

Число  $d$  играет важное значение в качестве работы алгоритма, малые значения  $d$  будут приводить к тому, что можем пропустить в модели учёт важных параметров, большие же значения будут приводить к переобучению. Можно доказать, что при  $d = \min(m, n)$ , произведение  $U \cdot V$  будет в точности равно целевой матрице  $R$ .

# Кросс-валидация для поиска оптимального значения

Используем отложенную выборку, на которой будем запускать алгоритм с различными значениями  $d$ , и выберем то, где RMSE ошибка будет минимальна.

