









Collaborative filtering

Рассмотрим матрицу взаимодействий “пользователь-товар”

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

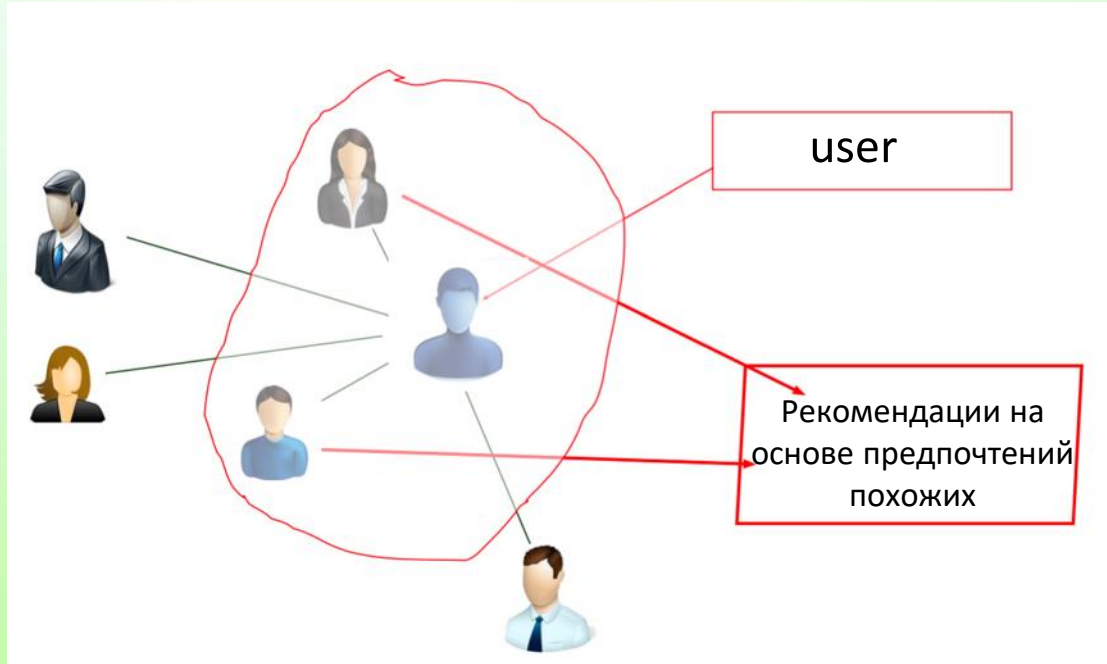
User-based CF

Как сделать рекомендацию для пользователя user?

Идея: найдем похожих на user пользователей и порекомендуем ему понравившиеся им товары.



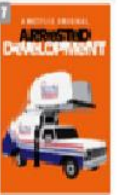




Такой подход называется user-based collaborative filtering.

User-based CF



Collaborative filtering

Какие фильмы рекомендовать выделенному пользователю?

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

?

Collaborative filtering

Найдем пользователей, смотревших те же фильмы

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

Collaborative filtering

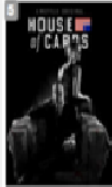








Найдем пользователей, смотревших те же фильмы

	4	5	6	7	8	9
						
	1	1	0		1	
	0	1	1			1
				1	1	0
		1	1		0	
		1				1

Похожие
пользователи

Collaborative filtering

Предложим нашему пользователю фильм, который он не смотрел, но смотрели похожие на него пользователи

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

Похожие пользователи

Collaborative filtering

Взаимодействие пользователя с товаром можно оценить не только по шкале “смотрел / не смотрел, купил / не купил”. Можно рассматривать оценки, выставленные пользователем товару.

Оценка:





- может быть **явной** (explicit): например, оценка фильмов по пятибалльной шкале, лайк.
- может быть **неявной** (implicit): например, факт просмотра товара/фильма, чтения поста в социальной сети.

Collaborative filtering

Рассмотрим матрицу оценок "пользователь-товар"

Пользователи

Понравится?

						
				4	5	
	5	3	4			1
	2		1		2	
	4	1		5		4
	2		4			2
		5		1		

Товары

Оценка



User-based CF

Как ещё можно посчитать похожесть пользователей?













User-based CF

Как ещё можно посчитать похожесть пользователей?

Корреляция оценок!

Пример User-based CF

user →













							
				4	5		NA
	5	3	4			1	
	2		1		2		
	4	1		5		4	
	2		4			2	
		5		1			NA

Нет общих оценок!













Пример User-based CF

							$\text{sim}(u,v)$
				4	5		NA
	5	3	4			1	0,28
	2		1		2		
	4	1		5		4	
 	2		4			2	
		5		1			NA













Пример User-based CF

							
				4	5		NA
	5	3	4			1	0,28
	2		1		2		-1
	4	1		5		4	
	2		4			2	
		5		1			NA

Пример User-based CF

							
				4	5		NA
	5	3	4			1	0,28
	2		1		2		-1
	4	1		5		4	1
	2		4			2	
		5		1			NA

Пример User-based CF

							
				4	5		NA
	5	3	4			1	0,28
	2		1		2		-1
	4	1		5		4	1
	2	1,77	4			2	
		5		1			NA

User-based CF

Минусы user-based:

- У большинства пользователей не так много оценок, что приводит к неуверенной оценке похожести пользователей.
- Оценки конкретного пользователя меняются во времени, поэтому при добавлении хотя бы одной новой оценки его похожесть на других пользователей может сильно измениться.

Item-based CF

Рассмотрим другой подход – item-based collaborative filtering.

Идея: К оцененным пользователем товарам найдем наиболее похожие на них и порекомендуем их пользователю.

Формулы аналогичны, получаются заменой строк на столбцы.



Item-based CF









Рекомендуем
похожие

user оценил

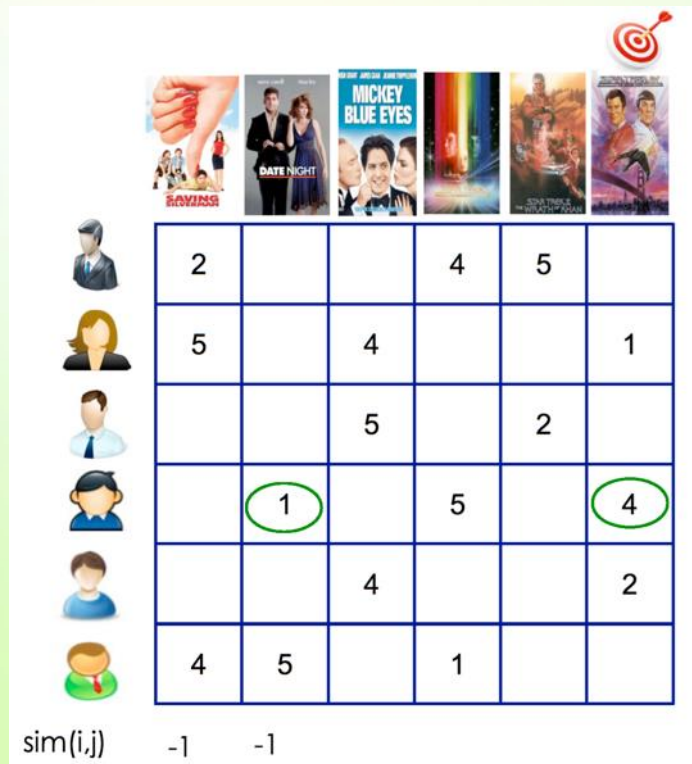
Пример Item-based CF



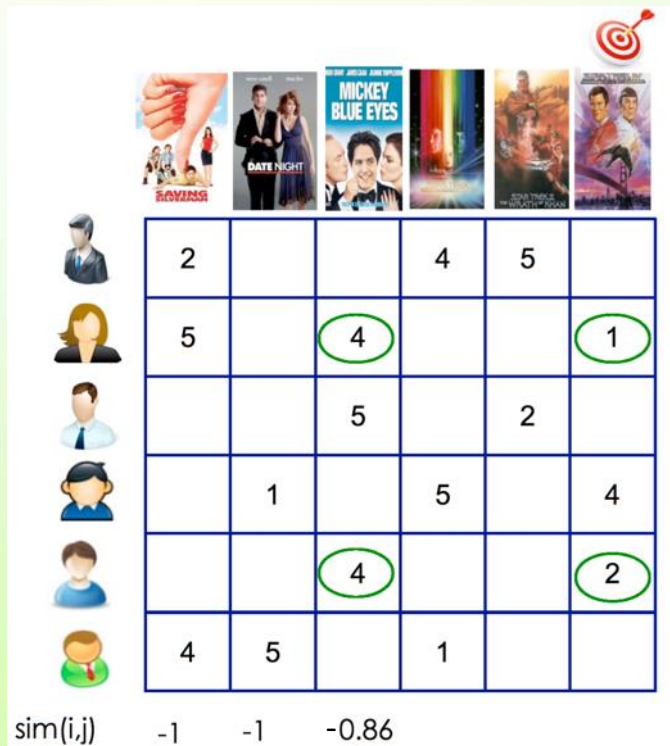
	2			4	5	
	5		4			1
			5		2	
		1		5		4
			4			2
	4	5		1		

sim(i,j) -1

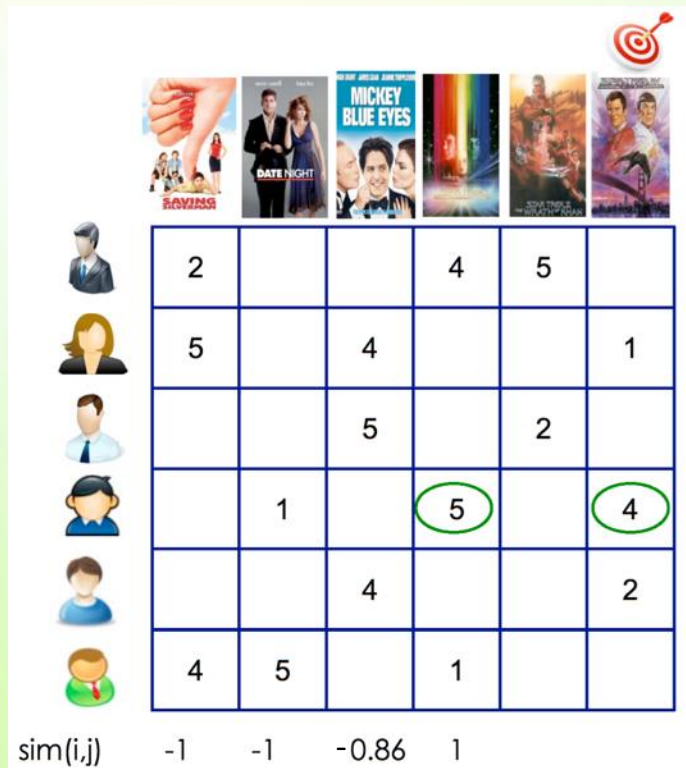
Пример Item-based CF



Пример Item-based CF



Пример Item-based CF



Пример Item-based CF



	2			4	5	
	5		4			1
			5		2	
		1		5		4
			4			2
	4	5		1		
$\text{sim}(i,j)$	-1	-1	-0.86	1	NA	

Нет общих оценок!

Item-based CF

Плюсы item-based:

- + Для популярных товаров можно получить надежную оценку похожести (много оценок юзеров).
- + Можно обновлять похожести товаров реже, например раз в день.

Обзор Collaborative Filtering

Плюсы:

- + Достаточно неплохие рекомендации при большом количестве явных оценок.

Минусы:

- Два пользователя должны оценивать одинаковые товары, оценка очень похожих товаров не учитывается в их близости.
- Проблема холодного старта: не знаем, что делать с новым товаром или пользователем.

Обзор Collaborative Filtering

Минусы:

- Сильная разреженность матрицы оценок приводит к плохим рекомендациям: например, вероятность того, что два пользователя, которые купили 100 книжек каждый, имеют хотя бы одну общую покупку (в каталоге из миллиона книг) равна 0.01 (в случае 50 покупок и 10 миллионов получаем 0.00025).

Content-based методы

Методы, основанные на вычислении похожести товаров или пользователей по их информации, не связанной с оценками, например,

- пол и возраст пользователя
- название и автор книги

Заменим корреляции оценок на эти похожести!

Далее как в CF

+ Помогает бороться с холодным стартом!

Новый товар, но знаем, что это «дорогой уют»

Пример Content-based

Похожесть книг по названию

Разложим название по словам

хороший банк
банк не понравился
понравился банк



хороший	банк	не	понравился
1	1	0	0
0	1	1	1
0	1	0	1

Пример Content-based

[illegible]

Пример Content-based

TFIDF Normed Vectors	a	Accelerating	and	applications	art	behavior	Building	Consumer	CRM	customer	data	for	Handbook	Introduction	Knowledge	Management	Marketing	Mastering	mining	of	relationship	Research	science	technology	the	to	using	website	your
Building data mining applications for CRM				0.502			0.502		0.344		0.251	0.502							0.251										
Accelerating customer relationships: using CRM and relationship technologies		0.432	0.296						0.296	0.216											0.468			0.432			0.432		
Mastering Data Mining: the art and science of Customer Relationship Management			0.256		0.374					0.187	0.187					0.256		0.374	0.187	0.374	0.256		0.374		0.374				
Data Mining your website											0.316								0.316								0.632	0.632	
Introduction to Marketing														0.636			0.436									0.636			
Consumer behavior						0.707		0.707																					
Marketing Research: a Handbook	0.537												0.537				0.368					0.537							
Customer Knowledge Management										0.381					0.736	0.522													

Похожи!

Векторы интересов

Решаем задачу рекомендации пользователям различных фильмов.

Можно описать пользователя и фильм векторами интересов:

- для пользователя – насколько он интересуется каждым жанром
- для фильма – насколько он относится к каждому жанру



Рейтинг

Будем определять **заинтересованность** как **скалярное произведение** вектора пользователя и вектора 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

У нас есть матрица рейтингов для задачи пользователь-фильм:

2	5	
5		4
	1	
	2	5

Цель: найти такие векторы пользователей и векторы фильмов, скалярное произведение которых максимально близко к рейтингам из таблицы.

Модели со скрытыми переменными

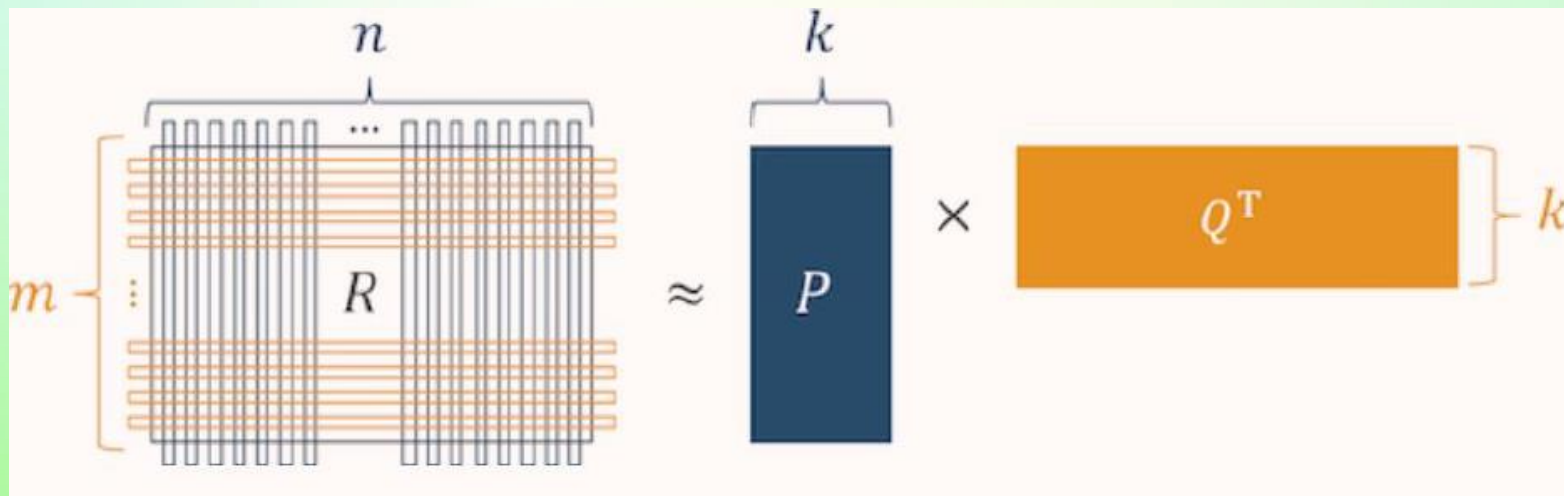
У нас есть матрица рейтингов для задачи пользователь-фильм:

	(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

Цель: найти такие векторы пользователей и векторы фильмов, скалярное произведение которых максимально близко к рейтингам из таблицы.

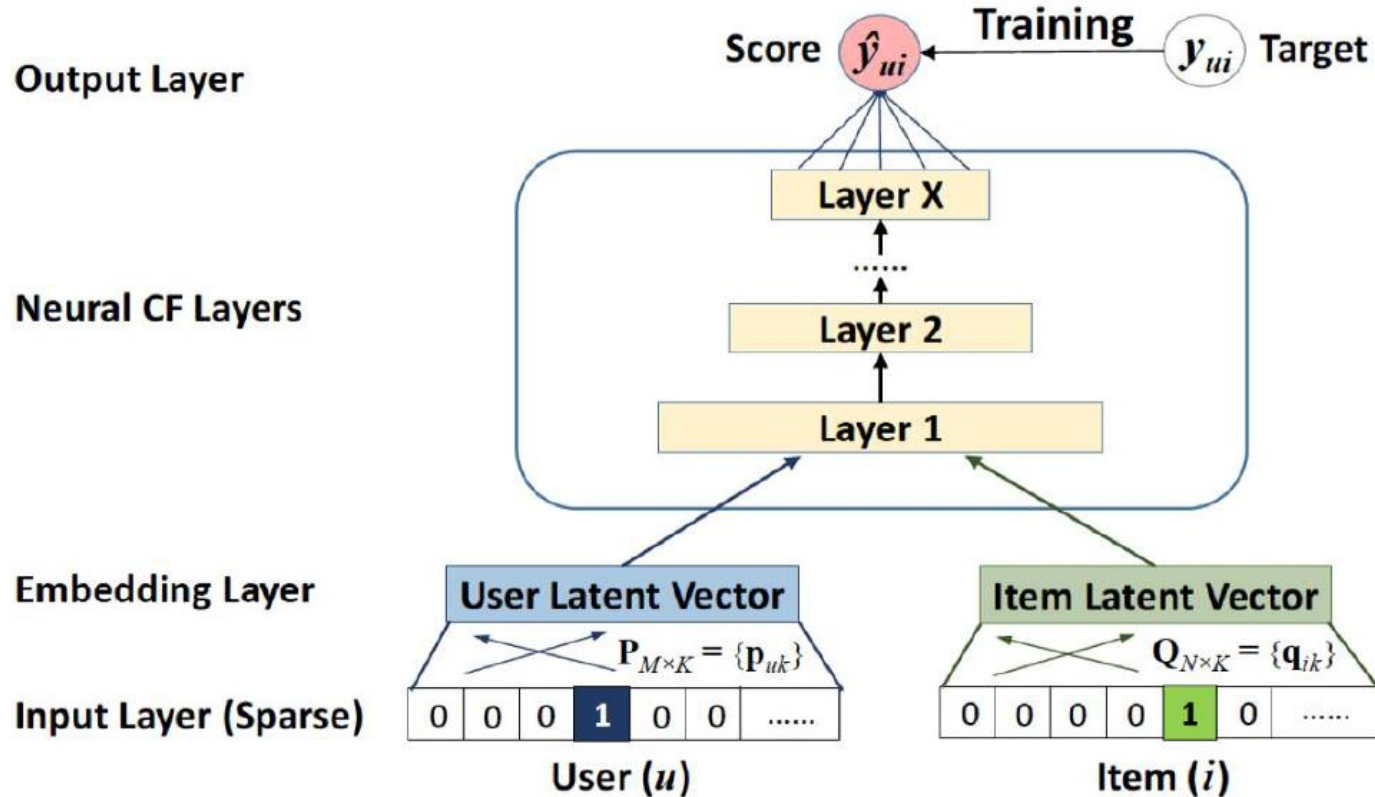
Матричные разложения

Эту задачу можно решить с помощью матричной факторизации, а именно, **представить матрицу рейтингов как произведение двух матриц**:



- в матрице P находятся векторы интересов пользователей
- в матрице Q находятся векторы фильмов

Neural collaborative filtering

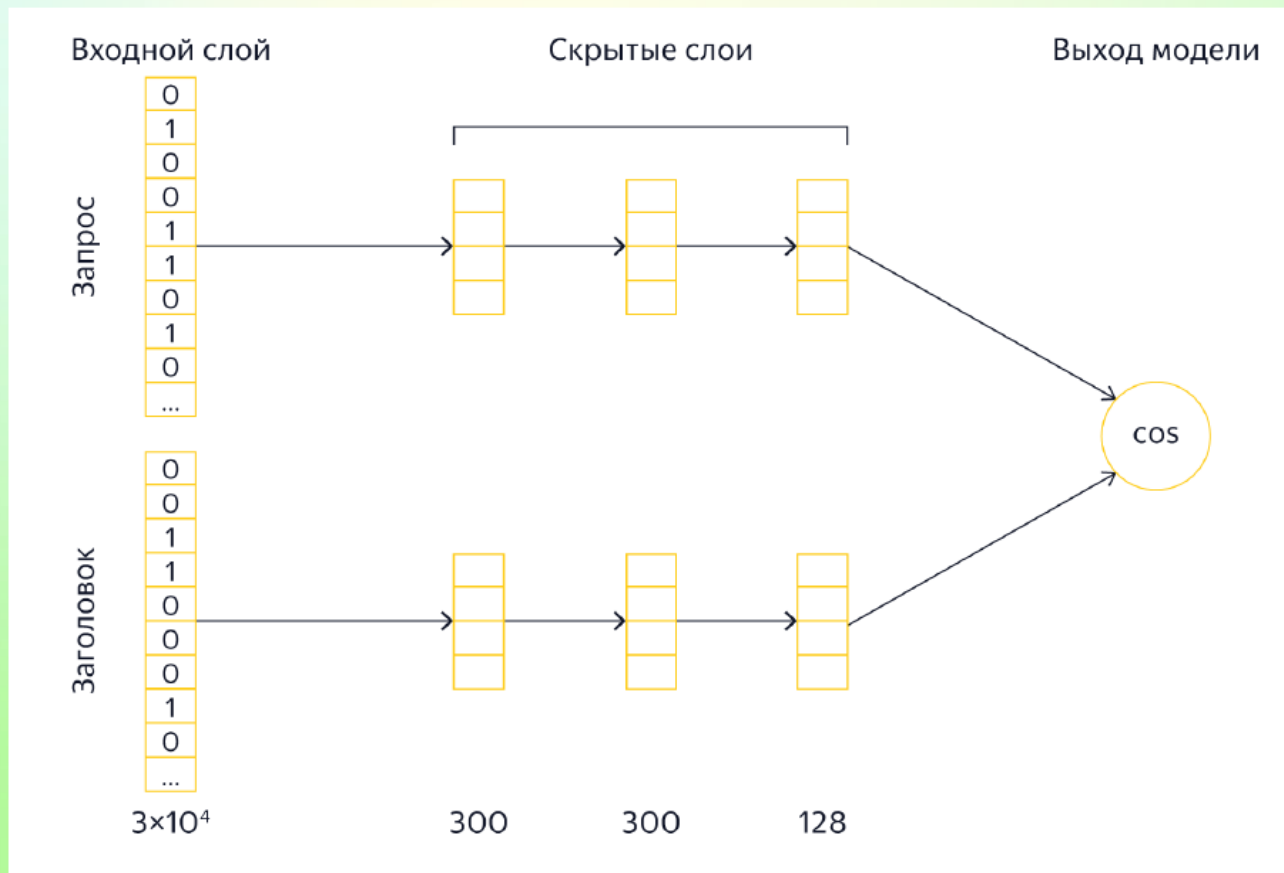


Deep Structured Semantic Model (DSSM)

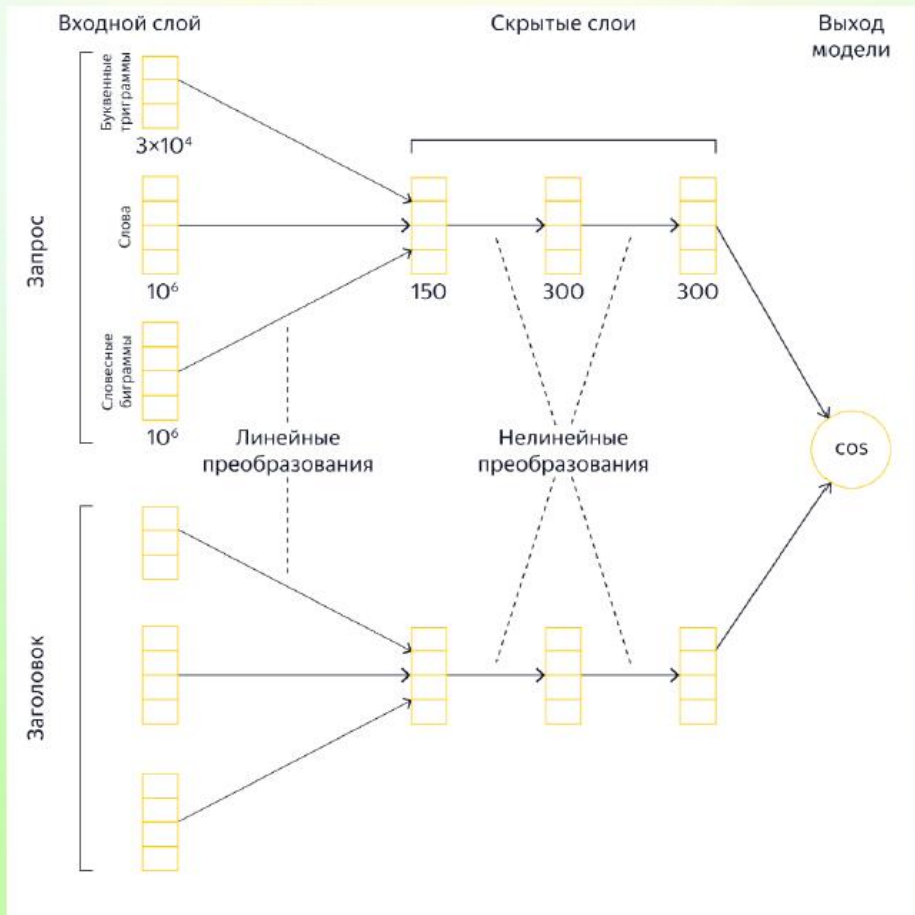
Предобработка текста (описание товара / описание пользователя):

- к тексту добавляются маркеры начала и конца
- затем текст разбивается на буквенные триграммы
- пример: палех -> [па, але, лех, ех]

Deep Structured Semantic Model (DSSM)



Deep Structured Semantic Model (DSSM)



Хорошие свойства рекомендательной системы

Какими свойствами должна обладать хорошая рекомендательная система?

Хорошие свойства рекомендательной системы

Diversity (разнообразие)



Born This Way

Lady Gaga



Pink Friday

Nicki Minaj



Dangerously in
Love

Beyoncé



Born This Way
– The Remix

Lady Gaga



Femme Fatale

Britney Spears



Can't be Tamed

Miley Cyrus



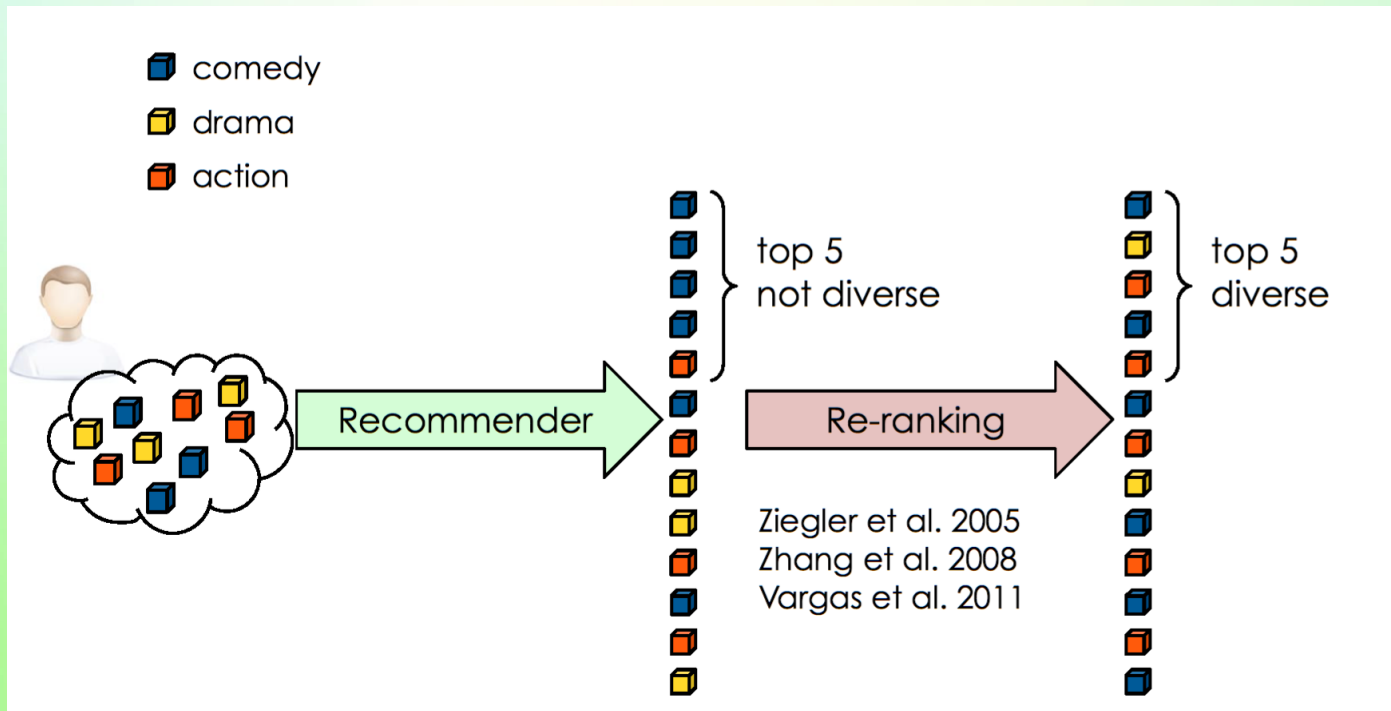
Teenage Dream

Katy Perry

Разные артисты

Хорошие свойства рекомендательной системы

Как можно сделать



Хорошие свойства рекомендательной системы

Novelty (новизна рекомендаций)

- Как оценить?
- Как улучшить?

Хорошие свойства рекомендательной системы

Serendipity (способность удивить)

- Порекомендовать Star Wars II тому, кто посмотрел Star Wars I — очевидно и бесполезно
- Как оценить?
- Как улучшить?

Хорошие свойства рекомендательной системы

Serendipity (способность удивить)

- Не рекомендовать слишком очевидные фильмы (все друзья это купили/посмотрели)
- Рекомендовать фильмы, которые максимально не похожи на предпочтения пользователя