Рекомендательные системы и не только ...

Jiji.ng

Киев, 2018

Содержание

Постановка проблемы

$u \in \mathbb{U}$ - множество пользователей

$i \in \mathbb{I}$ - множество товаров

$r_{ui} \in \mathbb{R}$- множество событий

- Offline models
  - Предсказать предпочтение

$$r'_{ui} = predict(u, i) \simeq r_{ui}$$

  - Персональные рекомендации

$$u \longmapsto (i_1 \dots i_k) = recommend_k(u)$$

  - Похожие объекты

$$u \longmapsto (i_1 \dots i_M) = similar_M(i)$$

Постановка проблемы

Постановка задачи звучит следующим образом: для каждого активного пользователя показать top-N объявлений с наибольшей вероятностью запроса контакта.

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
User 1	5	4	5			
User 2	4		5			
User 3		3	5		4	
User 4				3	4	
User 5			4	2	4	
User 6	3					5

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
User 1	5	4	5			
User 2	4	?	5			
User 3		3	5	?	4	
User 4			?	3	4	
User 5	?		4	2	4	?
User 6	3					5

Offline-модели

Offline-модели рекомендаций глобально делятся на коллаборативные и контентные. Очевидно, что каждая из этих моделей имеет свои плюсы и минусы и наилучшие результаты показывают гибридные модели, которые учитывают как историю действий пользователей, так и контент объявлений.

Content-based

- title
- description
- images
- ...

Algorithms: Latent Dirichlet allocation(LDA), relevance feedback(RF), TF-IDF

Оставим в векторах только те элементы, для которых нам известны значения в обоих векторах, т.е. оставим только те продукты, которые оценили оба пользователя, или только тех пользователей, которые оба оценили данный продукт. В результате нам просто нужно определить, насколько похожи два вектора вещественных чисел.

Подсчитаем коэффициент корреляции:

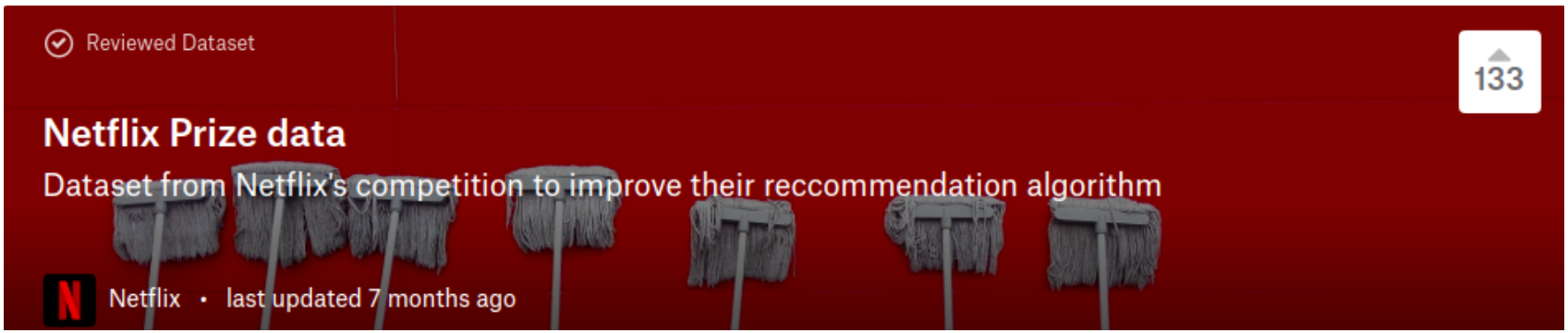
$$w_{ij} = \frac{\sum_a (r_{ai} - \overline{r_i})(r_{aj} - \overline{r_i})}{\sqrt{\sum_a (r_{ai} - \overline{r_i})} \sqrt{\sum_a (r_{aj} - \overline{r_j})}}$$

где,

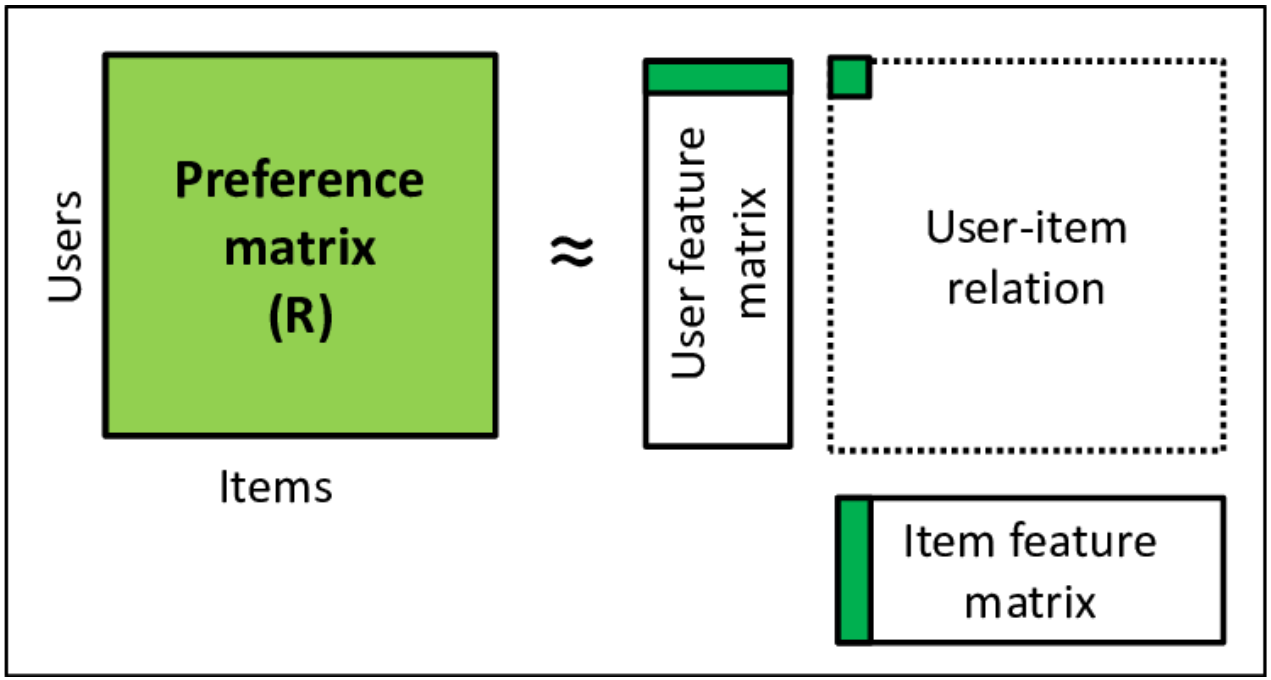
$\overline{r_i}$ - средний рейтинг, выставленный пользователем i

Matrix-factorization

On September 21, 2009, the grand prize of US 1,000,000 was given to the BellKor's Pragmatic Chaos team which bested Netflix's own algorithm for predicting ratings by 10.06

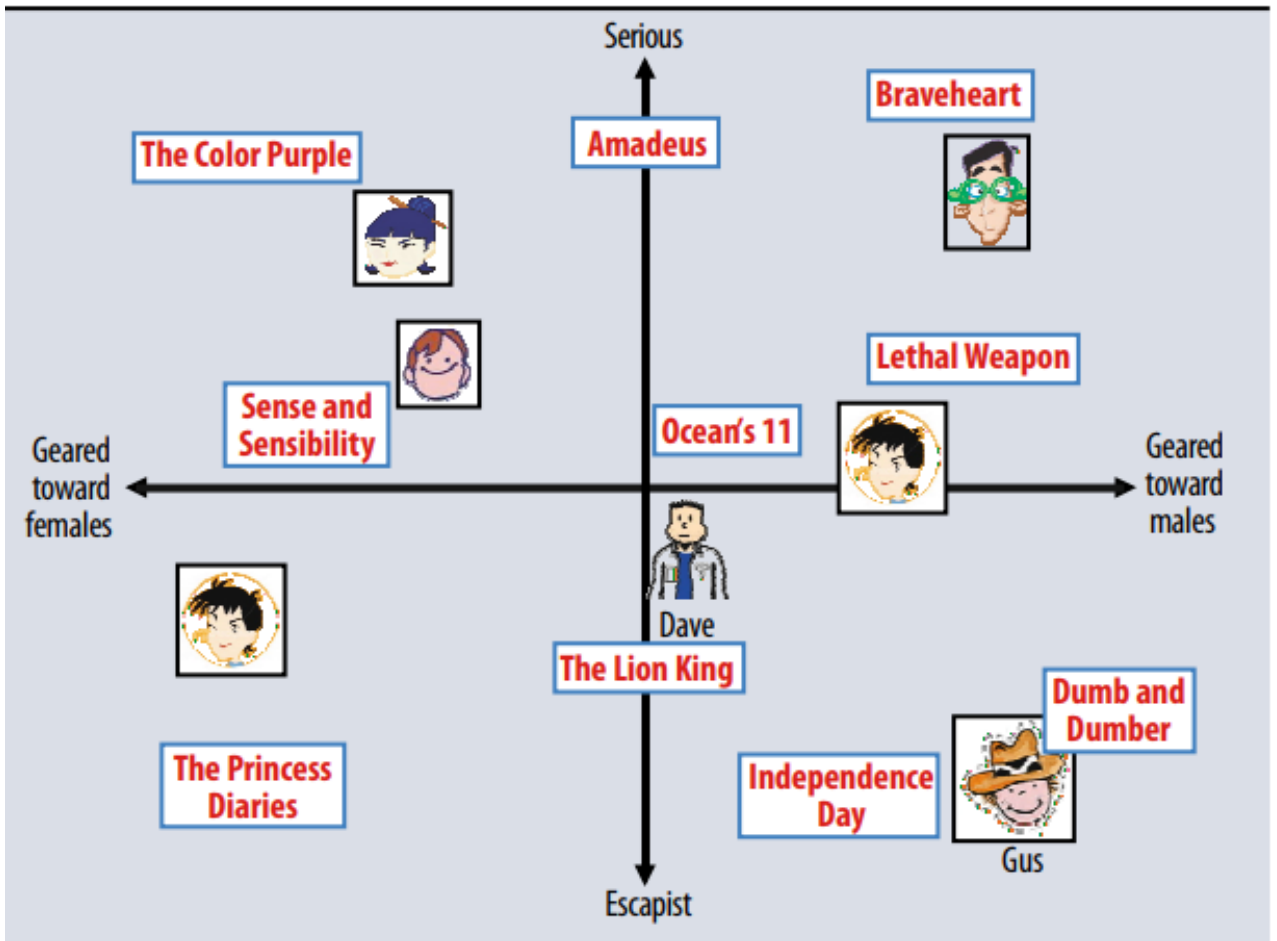


Latent Factor Models



Algorithms: Alternating least squares(ALS), Stochastic gradient descent(SGD)

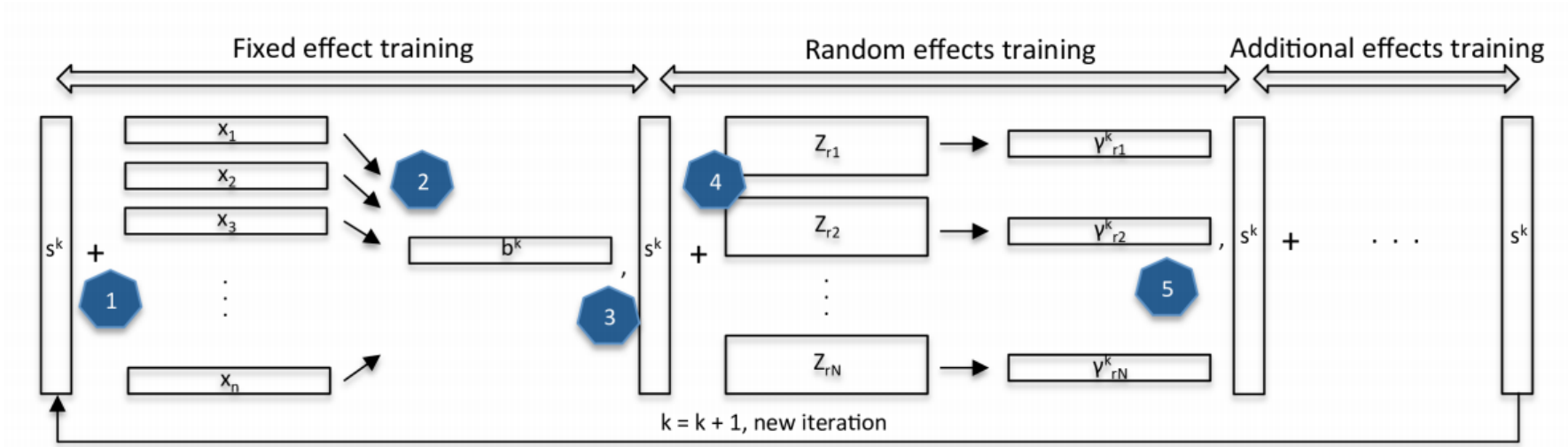
Matrix-factorization




LinkedIn

Photon-ml

- Generalized Linear Model (GLM)
- Generalized Additive Model (GAM)
- Generalized Additive Mixed-Effect Model(GAME)
- GLMix(Generalized Linear Mixed) = GLM + per-user model + per-item model



The experiments were conducted on a cluster consisting of 135 nodes managed by Apache YARN 3. Each node has 24 Intel Xeon(R) CPU E5-2640 processors with 6 cores at 2.50GHz each, and every node has 250GB memory.

Evaluation

Academic metrics:

- RMSE
- MAE
- Precision/Recall

(all may have low correlation with actual user satisfaction)

Business metrics:

- CTR/CVR
- ROI
- CLV (Customer Lifetime Value)

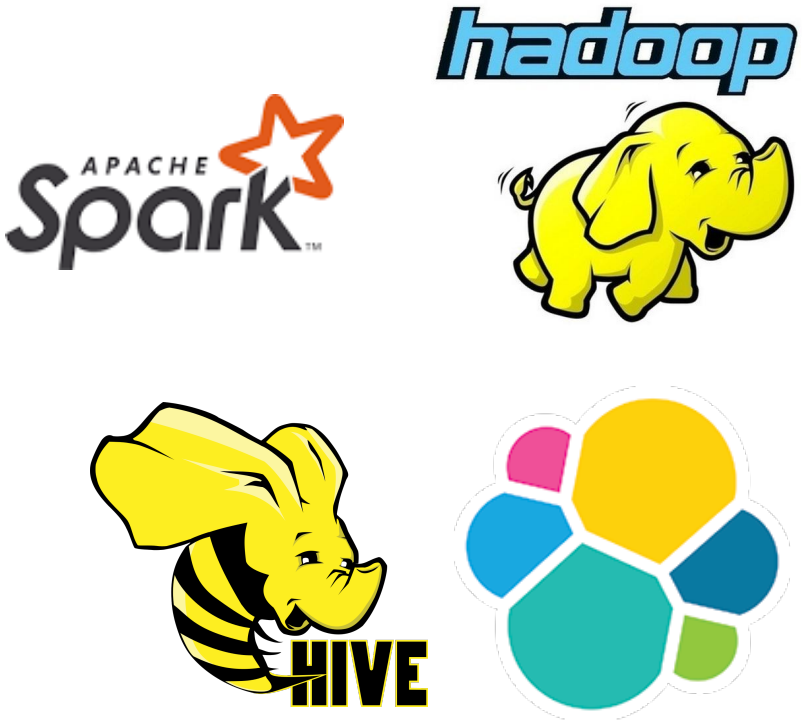
Customer metrics:

- Coverage – covering more items for recommendations
- Diversity – higher variety of items (rich-get-richer effect)
- Novelty – recommending new items

Проблемы

- Товары быстро продаются, не успев даже набрать хорошую историю по просмотрам и запросам контактов. Классические алгоритмы коллаборативной фильтрации устроены так, что объявления с короткой историей не попадают в рекомендации. Чаще рекомендуются долго живущие объявления, которые, как правило, представляют меньший интерес для покупателей.
- Проблемы холодного старта
- Как заставить это все быстро работать

Проблемы



Спасибо за внимание!