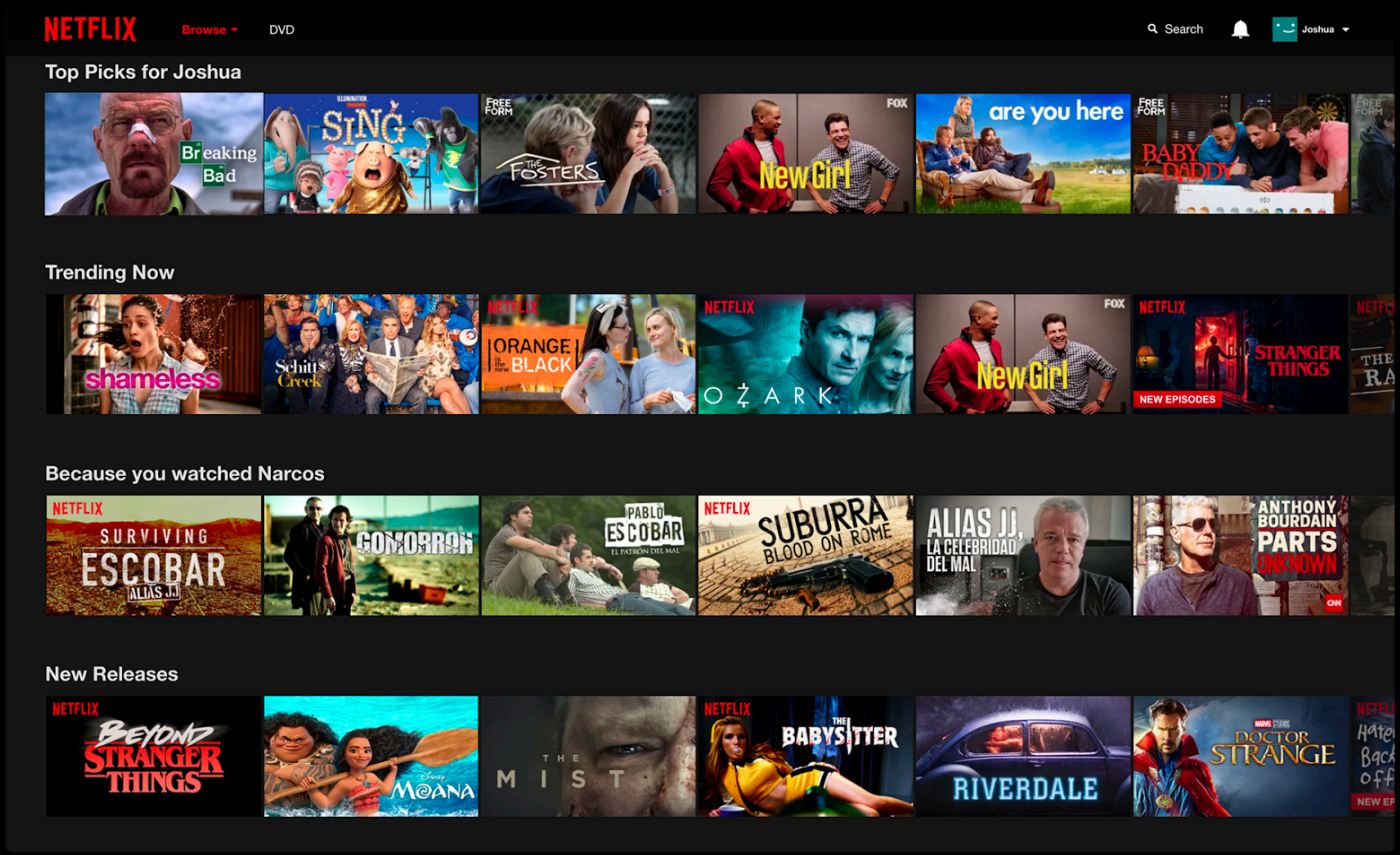
Recommendation Systems

Diversity. Debiasing.

Eugeny Malyutin / Sergey Dudorov

Recommender's result list

- 

Balance

Recommender’s result list

- Relevance
- Novelty
- Diversity
- Selected/Paid items

What is diversity?

- Diversity has been introduced as one of possible solution of over-fitting
- We have a lot of definitions of diversity
- Diversity is actual problem for each project

Diversity definitions

- Average dissimilarity for each pair of items in the result list

$$D = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \left(1 - \textbf{Similarity}\left(c_i, c_j\right)\right)}{n/2 * (n-1)}$$

- Diversity is represented as the Gini coefficient - a measure of distributional inequality.

$$G = 1 - 2\int_0^1 L(u)du$$

Diversity definitions

- Diversity is part of the calculation of the nDCG measure - it has a direct impact on the calculated probability value.

$$G[k] = \sum_{i=1}^{m} J\left(d_k, i\right)(1-\alpha)^{r_i,k-1}$$

- Diversity between two items is the product the item's relevance, similarity and places in the ranked list.

$$ILD\left(i_k \mid u, R\right) = C_k' \sum_{l} \mathrm{disc}(l \mid k) p\left(rel \mid i_l, u\right) dist\left(i_k, i_l\right)$$

Diversity definitions

- Diversity is a combination of genre coverage (how many different genres are present in the ranked list) and non-redundancy (genres do not repeat on the list).

$$\textbf{BinomDiv}\,(R) = \textbf{Coverage}\,(R) * \textbf{NonRed}\,(R)$$

- Ask your users - too expensive

Diversity Algorithms

- Improving Recommendation Lists Through Topic Diversification

1. Generate predictions (at least 5N for a final top-N recommendation list).
2. For each N+1 position item calculate the ILS (diversity) if this item was part of the top-N list.
3. Sort the remaining items in reverse (according to ILS rank) to get their dissimilarity rank.
4. Calculate new rank for each item as $r = a * P + b * P_d$, with P being the original rank, P_d being the dissimilarity rank and a, b being constants in range 0, 1].
5. Select the top-N items according to the newly calculated rank.

- A Content Recommendation System Based on Category Correlations

1. Collect genre information for each film in the dataset.
2. Calculate genre correlation for all film in the dataset by counting the number of occurrences of each possible pair of genres.
3. Normalize genre correlation values.
4. Collect user genre preferences explicitly.
5. Generate recommendations by calculating each predicted ratings as:
   5.1 $\Sigma R_M{}^* r_{ij}$ with R_M being the films average ratings and r_{ij} being the genre correlations of all genres liked by user i and being attributed to film j.
   5.2 Normalize the sum according to the number of user's genre preferences.
6. Select the top N items with the highest predicted rating and present them to the user.

Diversity Algorithms

- Temporal diversity in recommender systems

1. Generate recommendations using the Matrix Factorization, kNN or mean average rating approach.
2. Track if the user selected/rated any of the recommended items.
3. Generate a new recommendations using updated information (all new ratings provided by the user).
4. Calculate the diversity of the newly generated list of recommendations, by comparing it with the previous iteration (the list of recommendations presented to the user during his/her previous interaction with the system).

- Who likes it more?: mining worth-recommending items from long tails by modeling relative preference

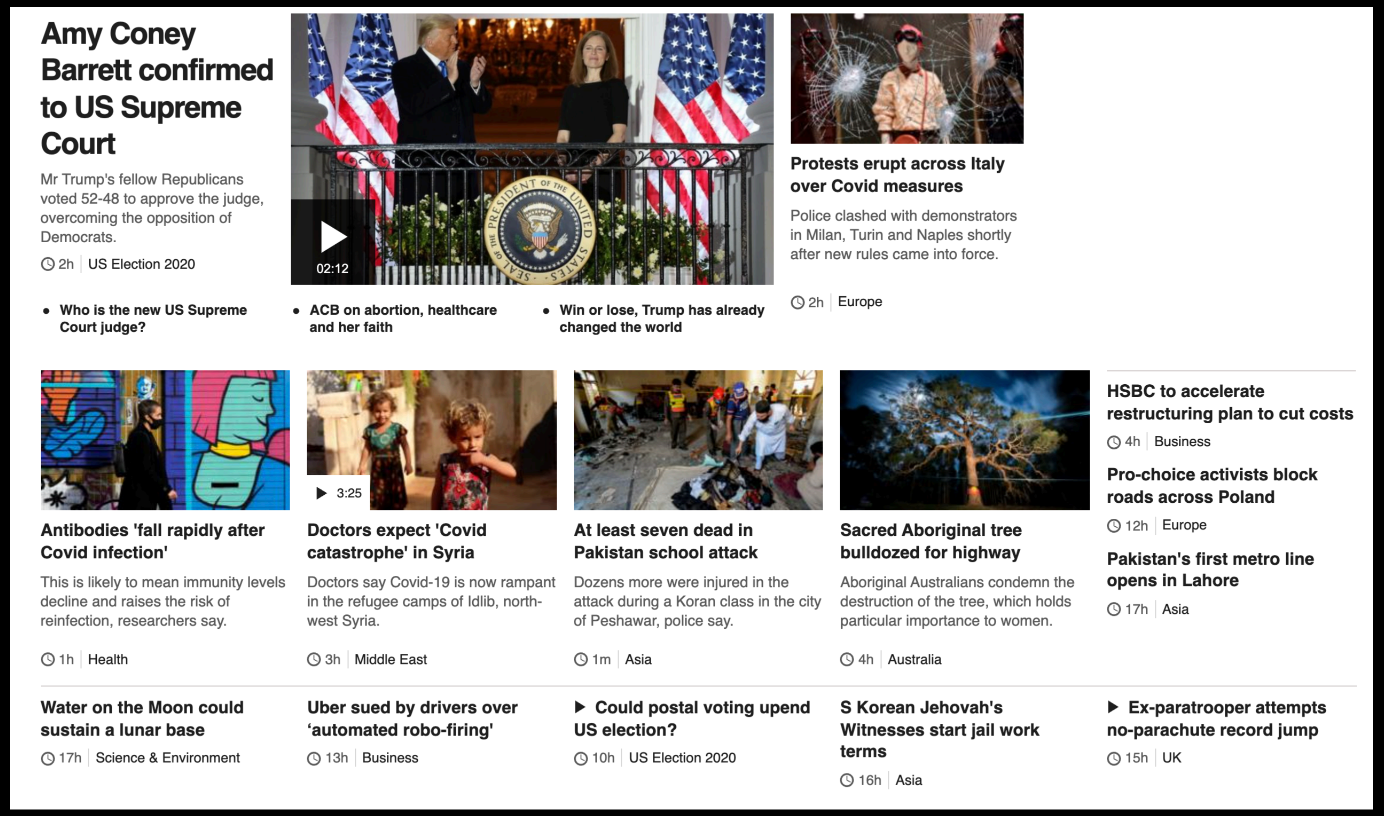
1. Create recommendations for all existing user using any CF approach.
2. For each item that was recommended at least once, calculate a 5D score as a combination of: Accuracy, Balance, Coverage, Quality and Quantity of long-tail.
3. For each user create a list of possible recommendations and:
   3.1 Order the list by predicted rating in order to assign a rank r to each item.
   3.2 Reverse order the list by each items 5D score in order to assign a rank r_{5D} to each item.
   3.3 Create a combined rank $r_n = r * r_{5D}$ and order the items by this rank.
4. Present the top N items to the user.

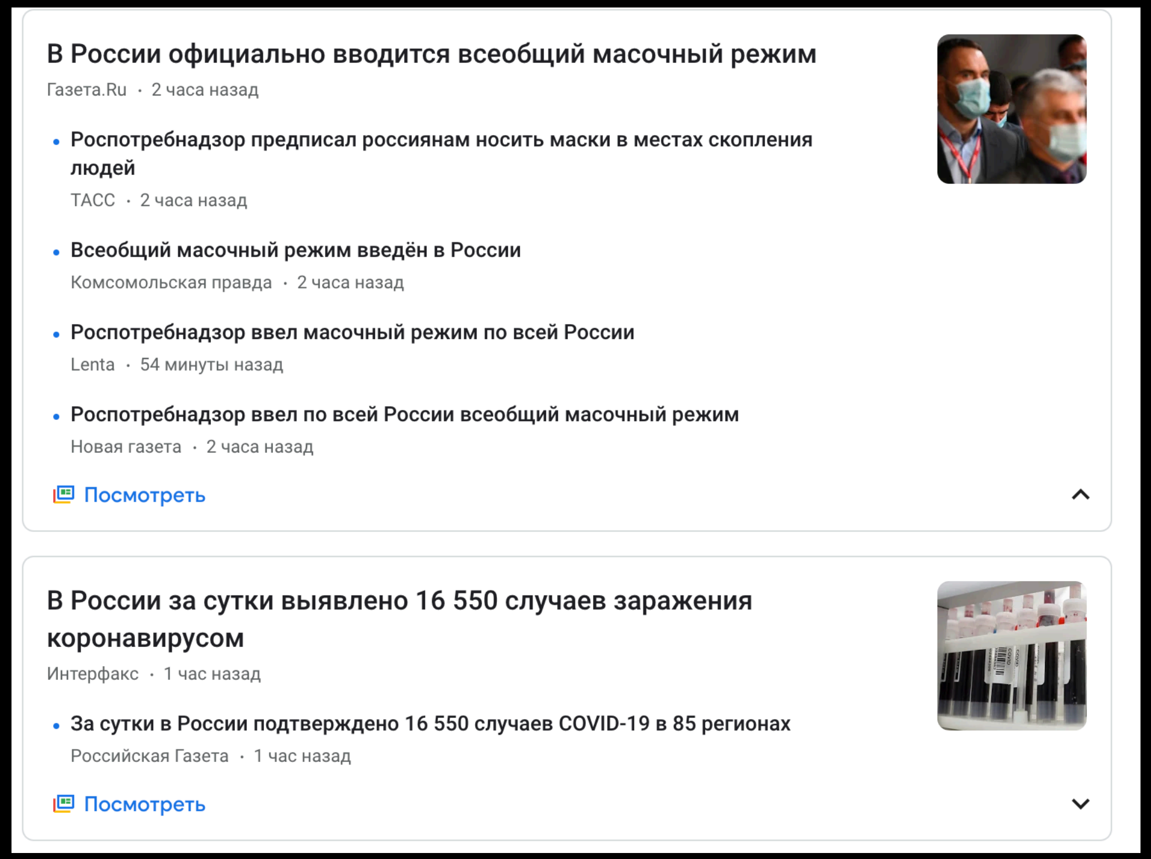
Diversity results

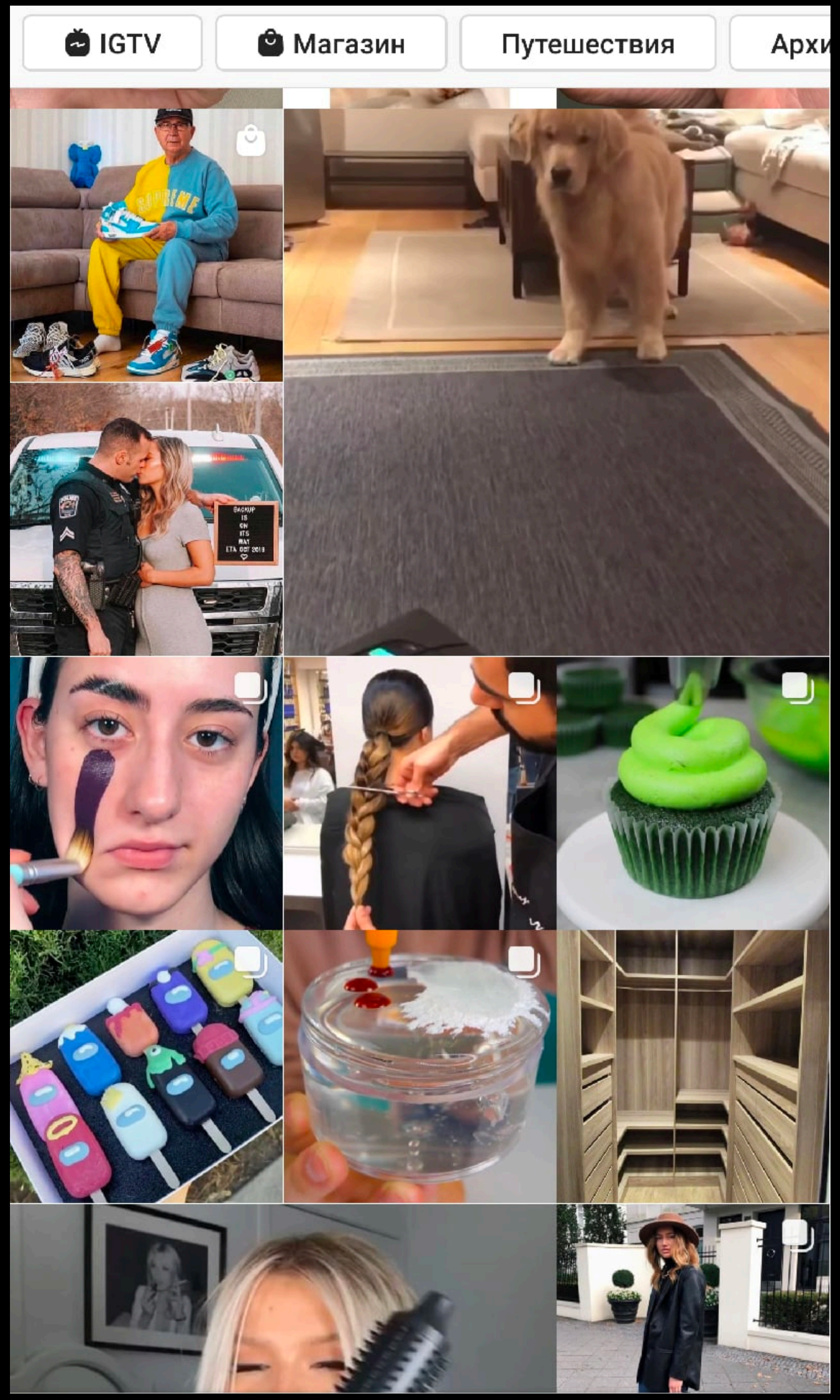
- Diversity has strong connection with domain
- Optimal diversity increase user’s satisfaction
- Diverse items better works in blocks
- Explanation improves user’s feedback

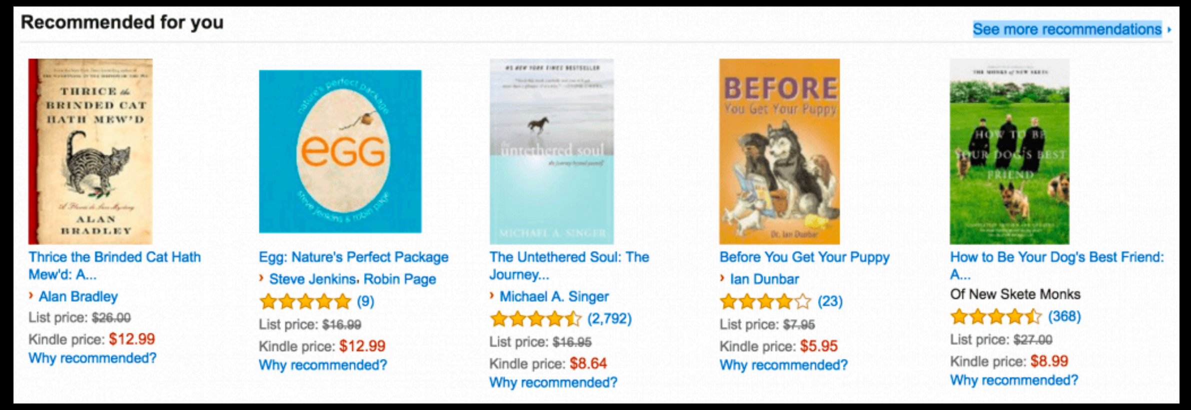
Novelty

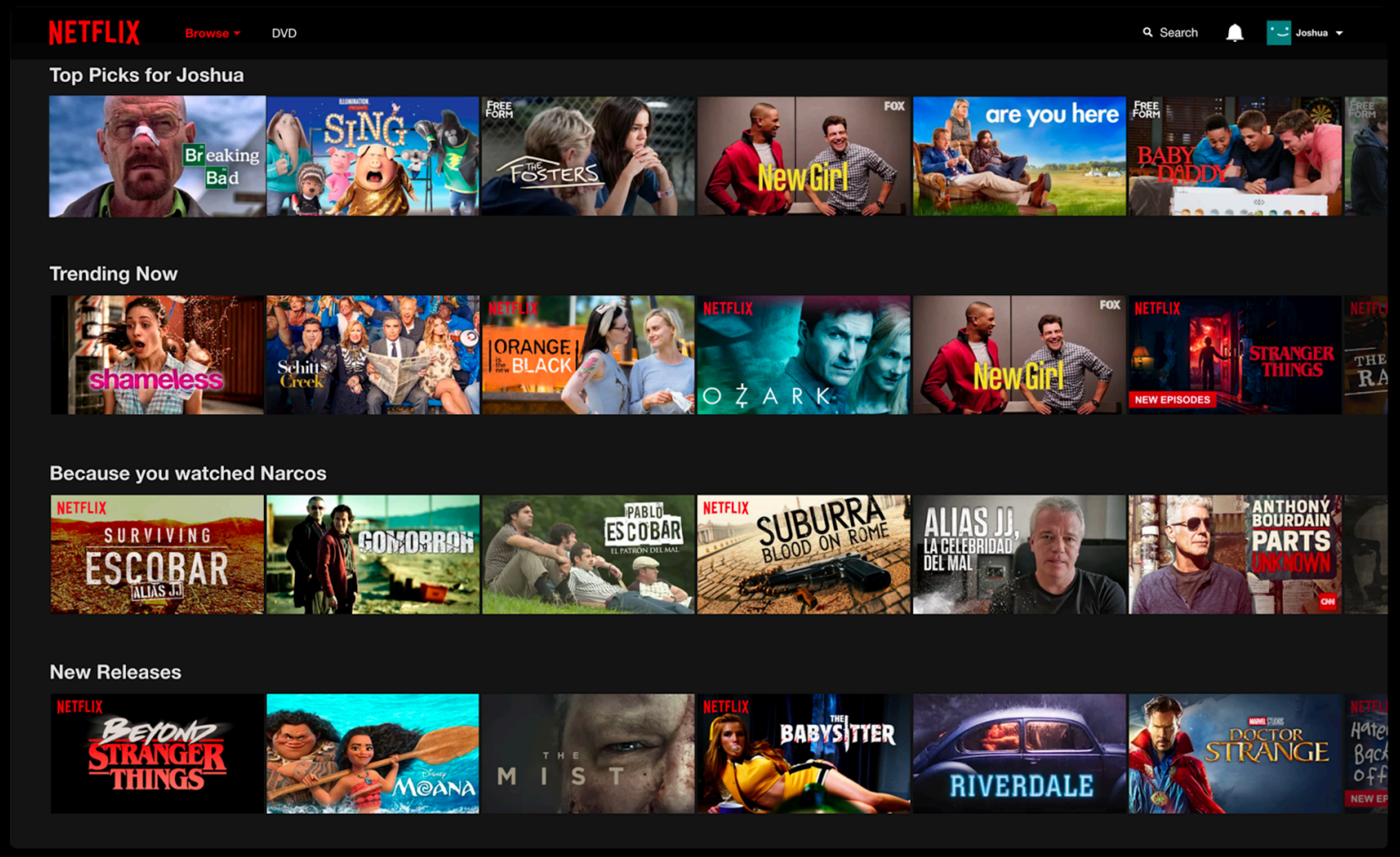
- Fresh content is always better
- Fresh content is important part
- Fresh content doesn't really matter









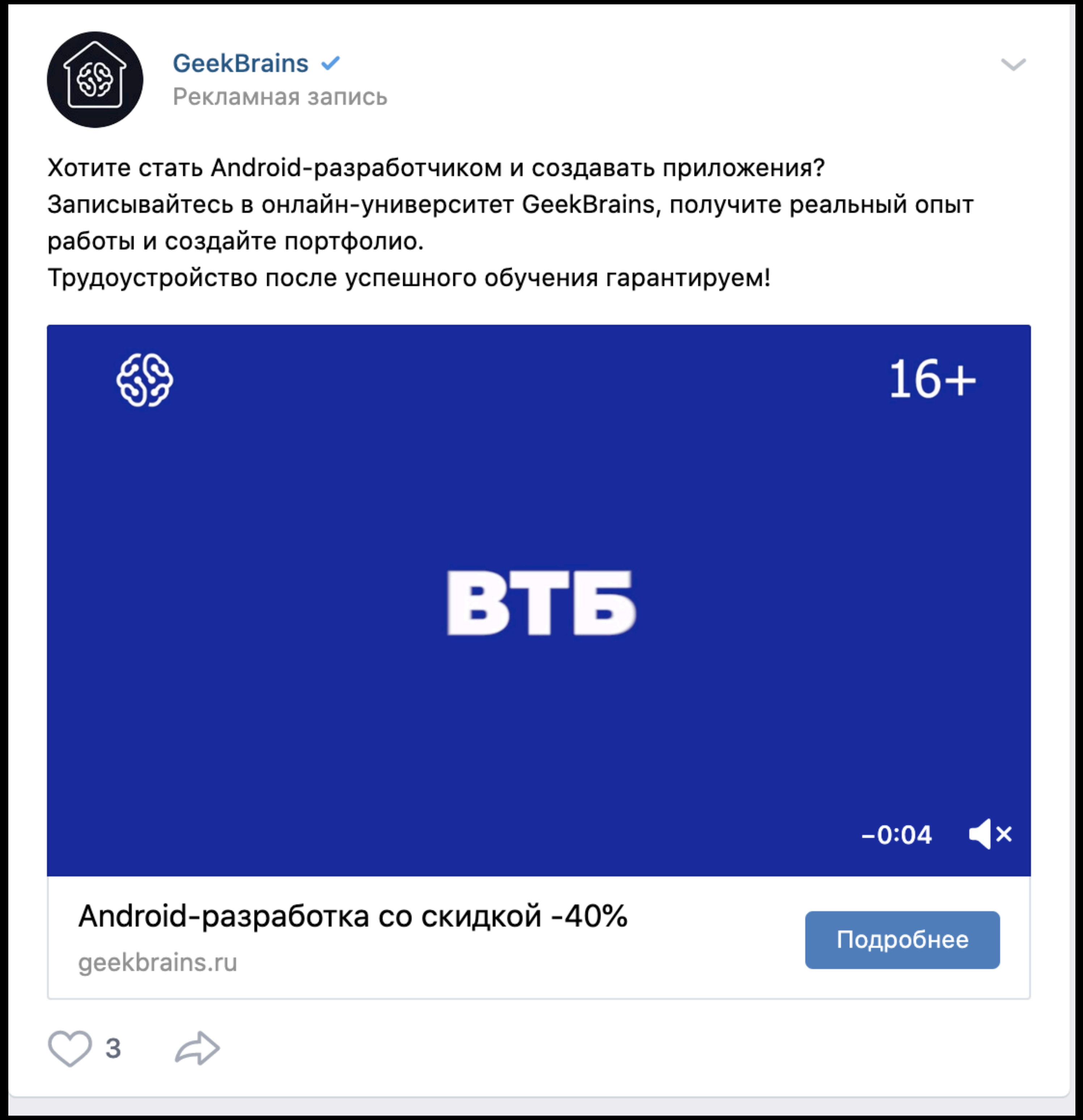


Paid items

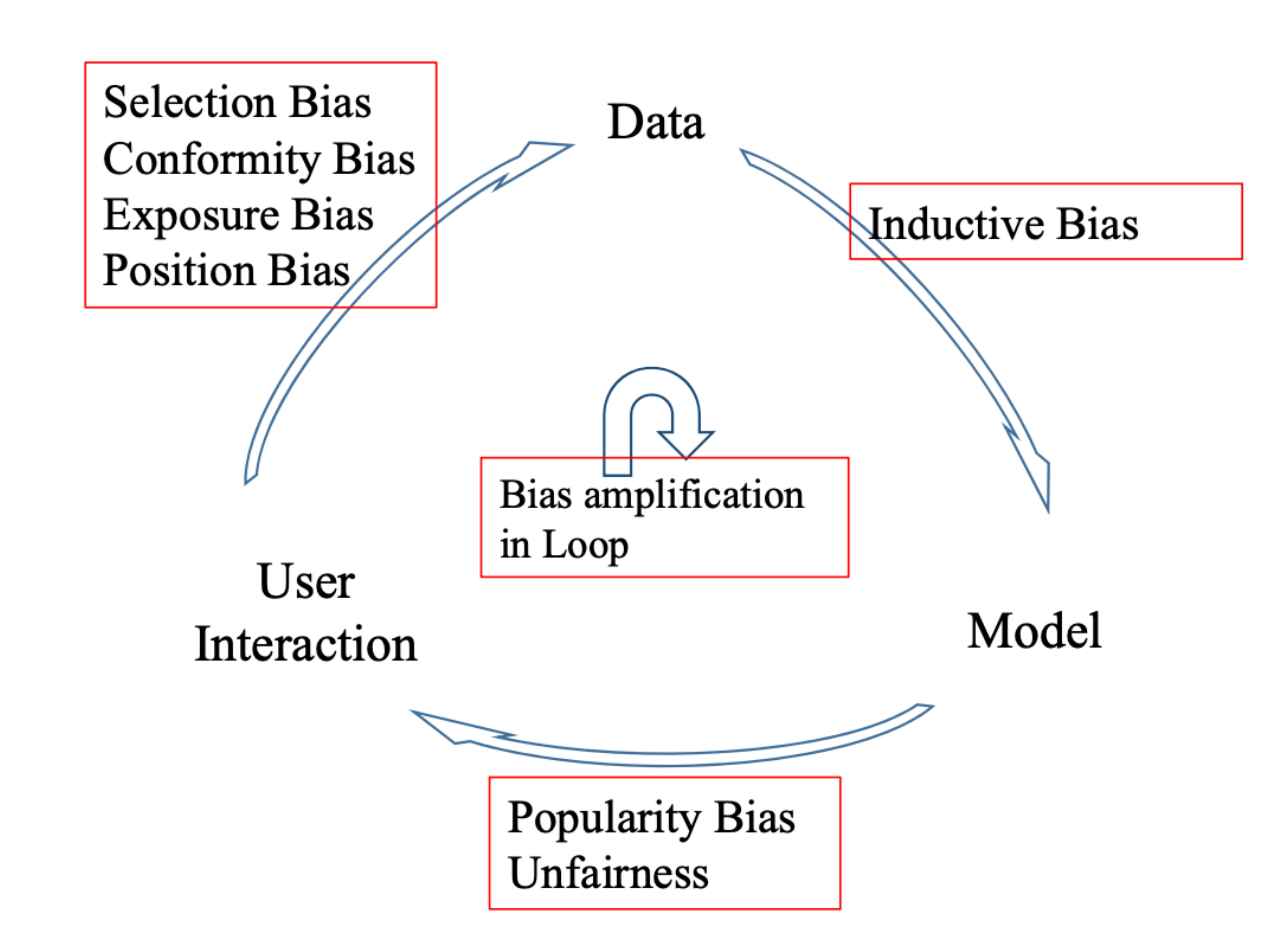
- Find relevance adv/paid items

- Use fix position for them

- Use dynamic position(active users watch more adv/paid items)



Bias

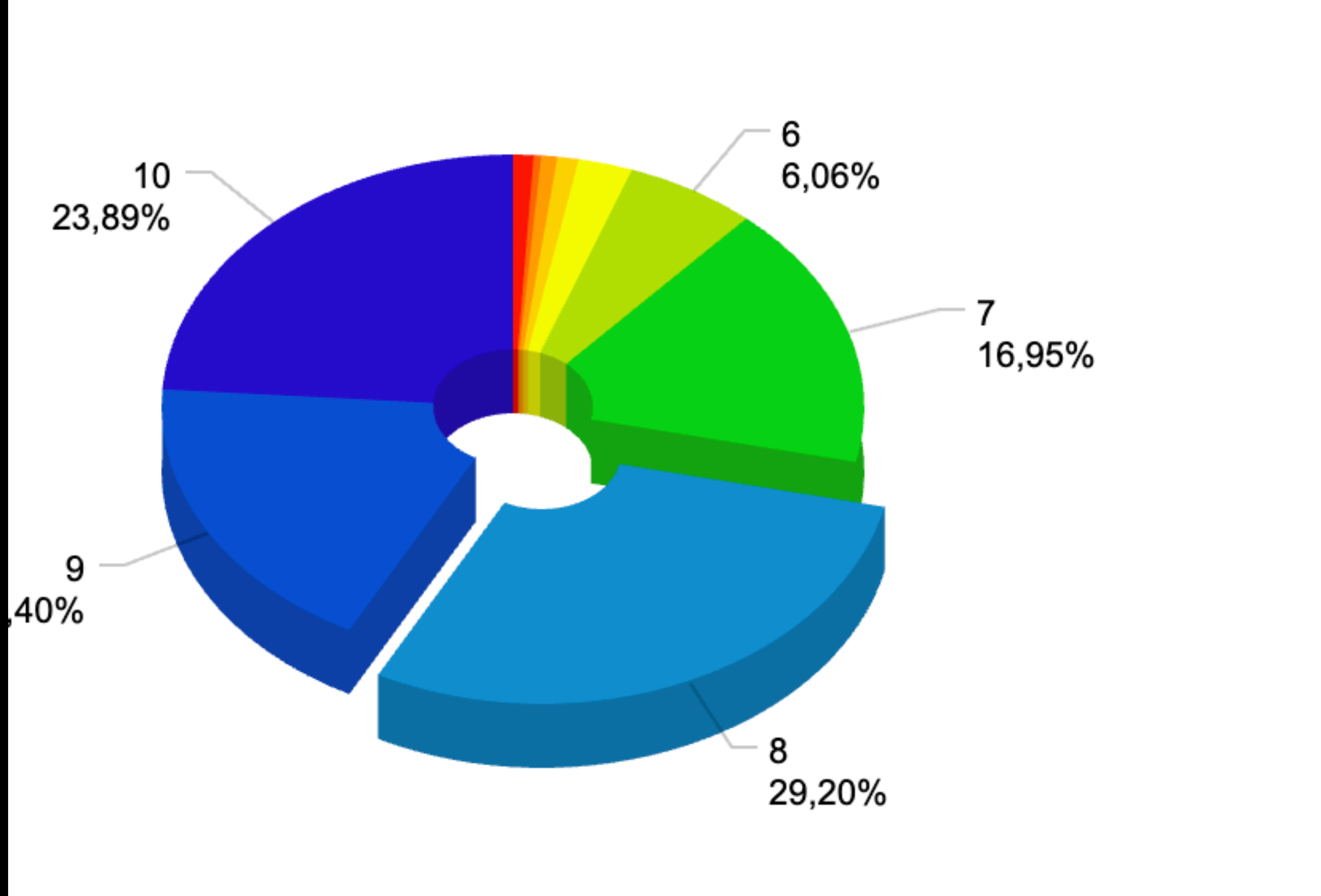

Selection Bias
Conformity Bias
Exposure Bias
Position Bias
Data
Inductive Bias
Bias amplification
in Loop
User
Interaction
Model
Popularity Bias
Unfairness

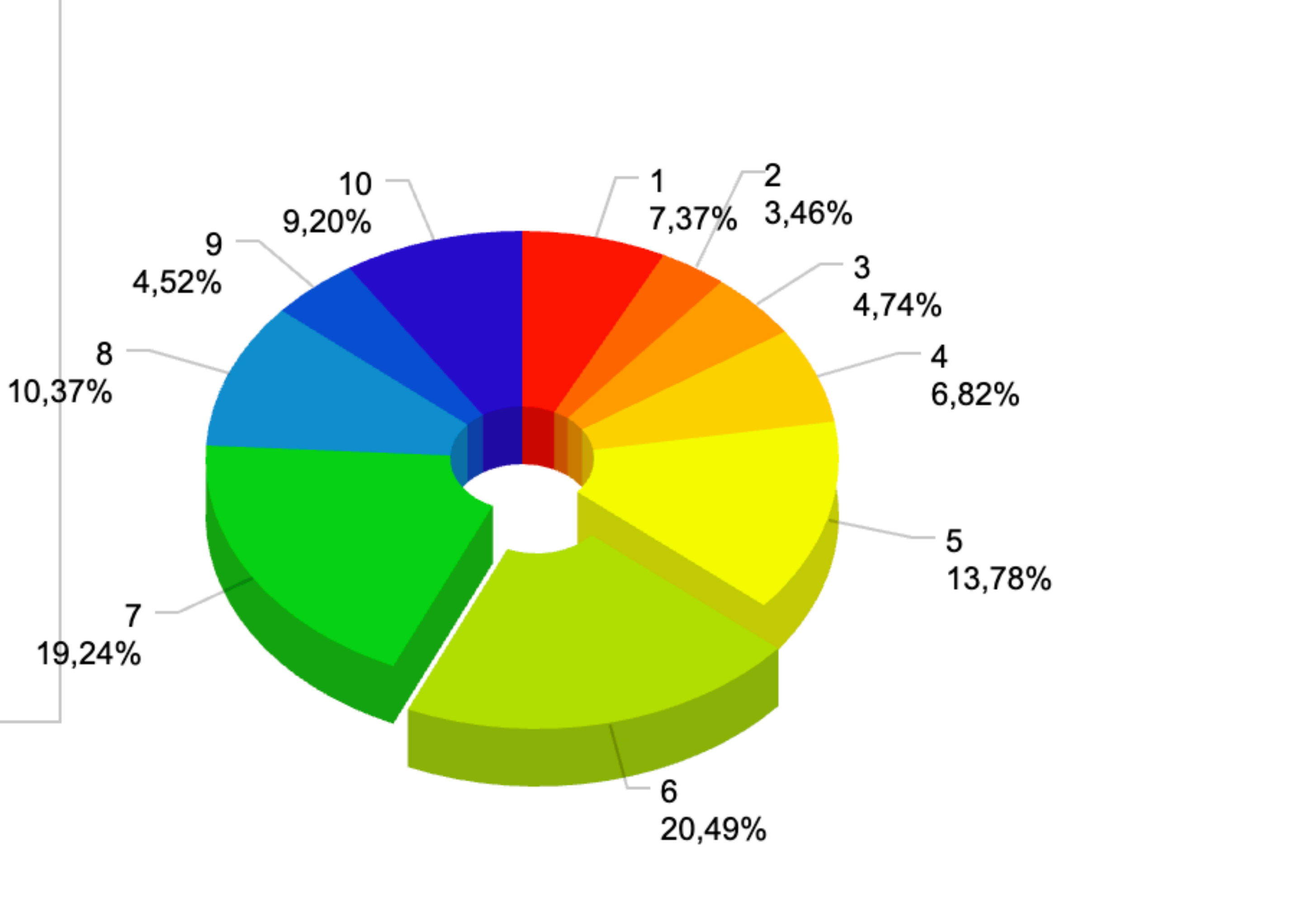
Selection Bias

- Only Explicit Feedback
- User prefer mark best and worst items

Conformity Bias

- Only explicit feedback
- Users rate item similarly to the others users





Exposure Bias

- User very biased on our recommendation lists

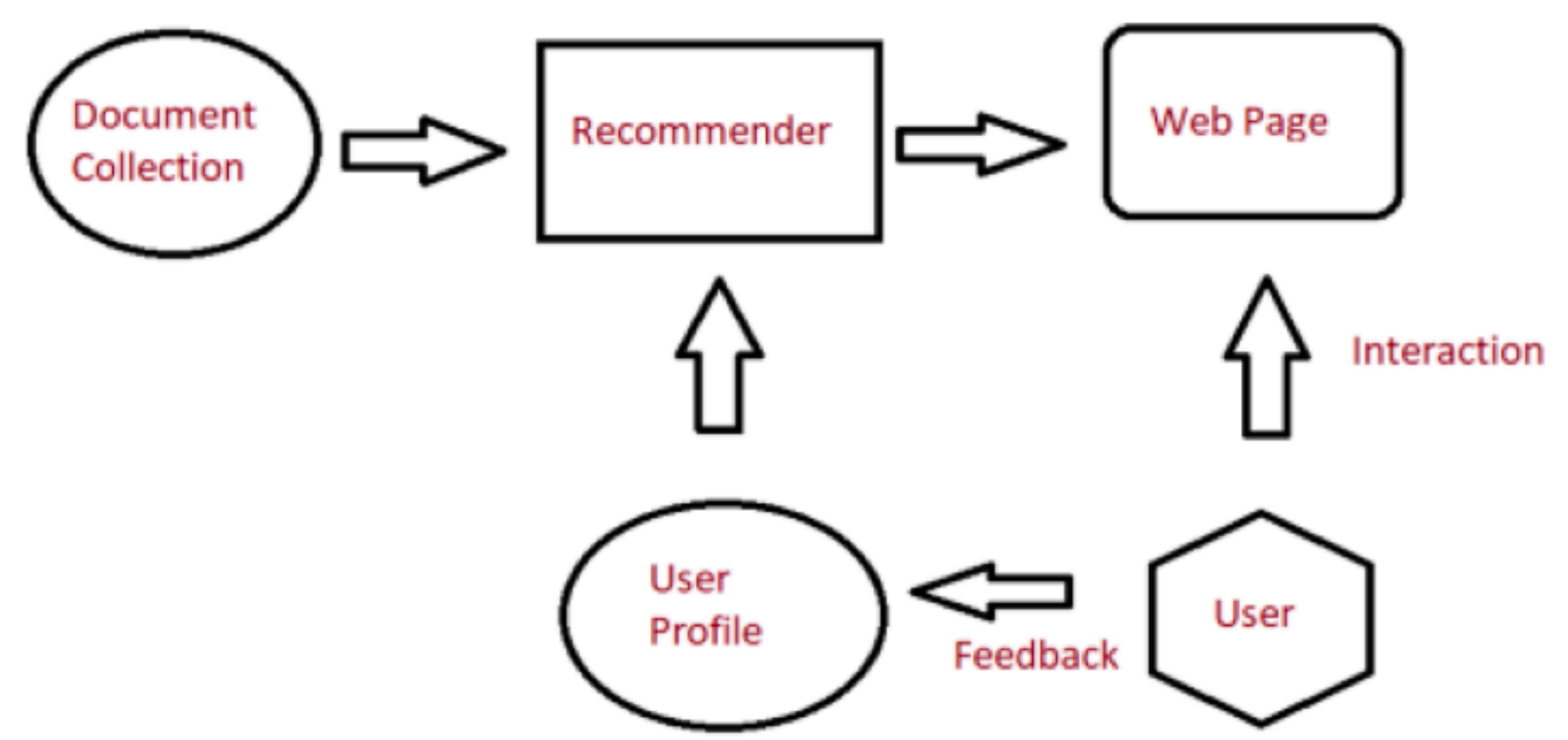


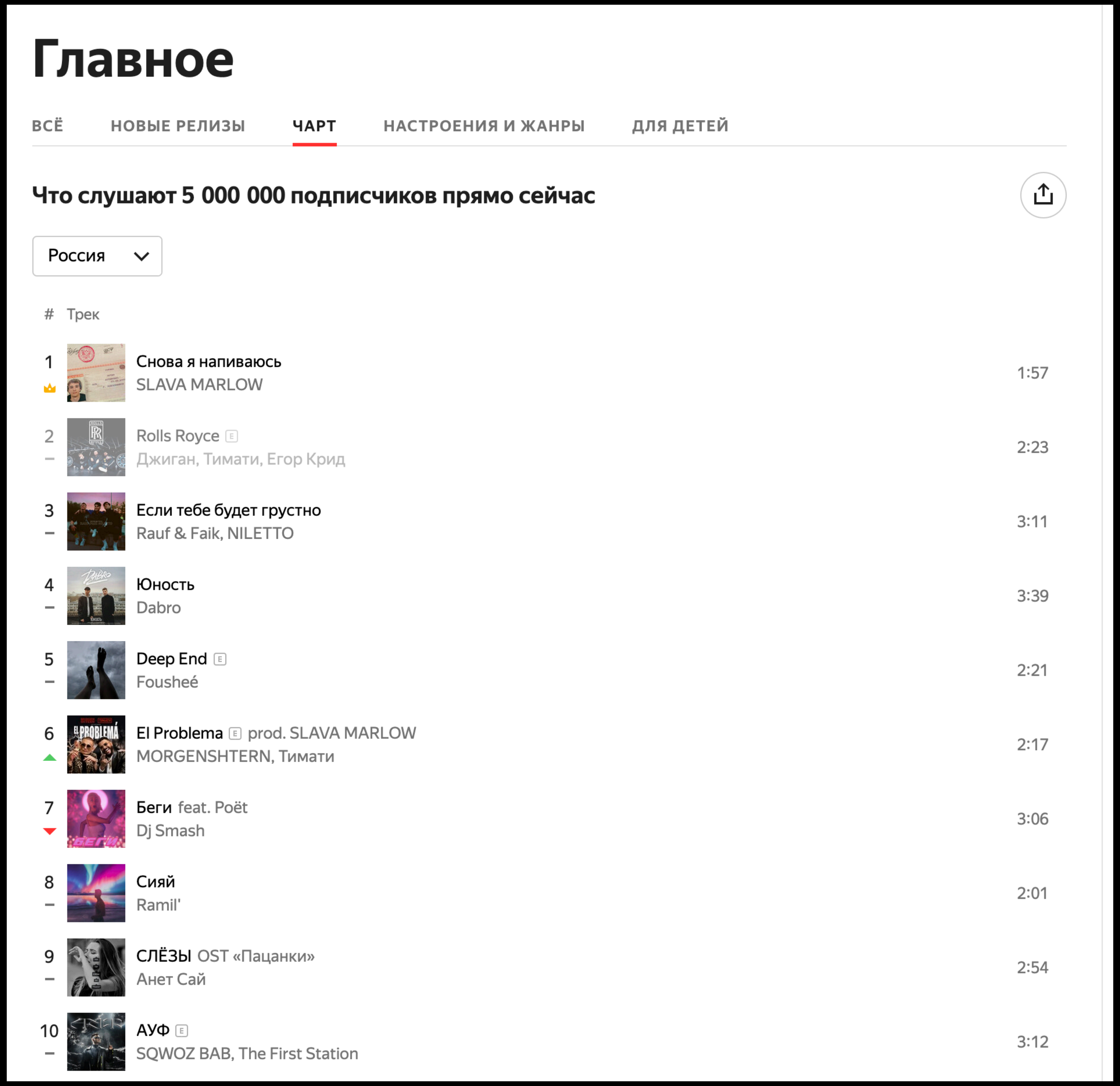
Fig: Recommender System

Position Bias

- Works on both types of feedback

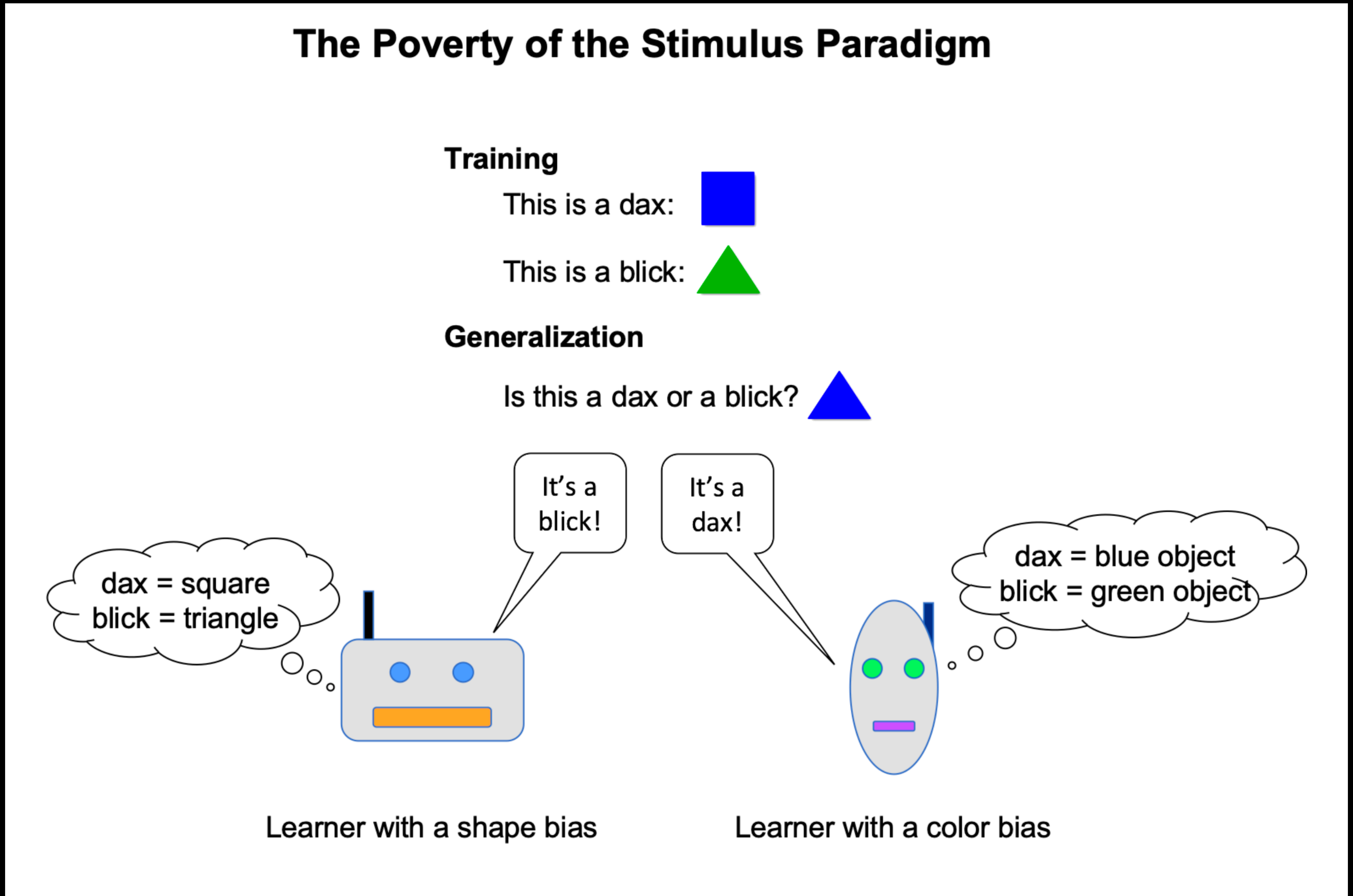
- Users prefer interact with items in higher position of the result list

- Lower items can be more relevant but user didn’t watch them



Inductive Bias

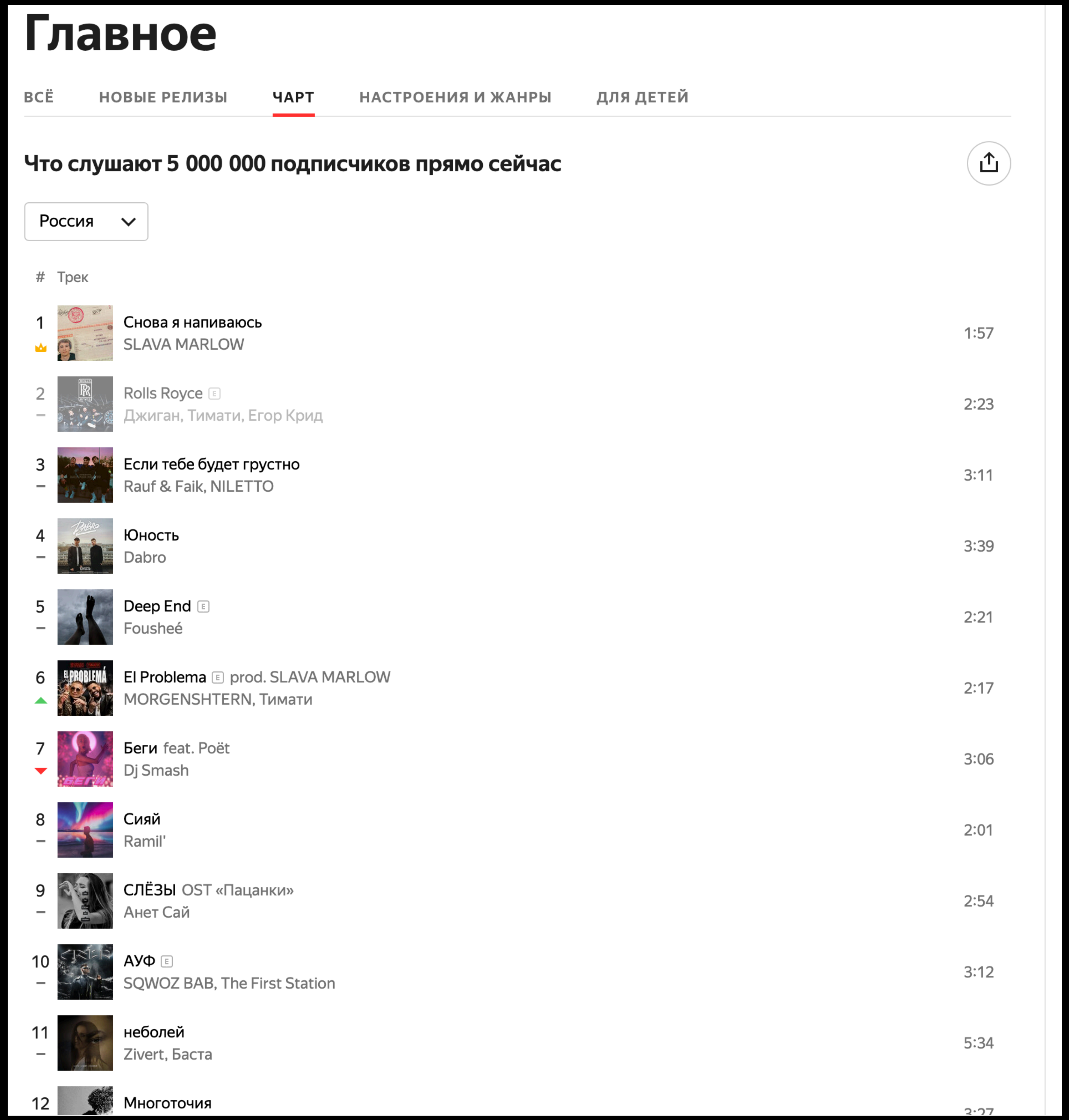
- What is your target?
- Can item hack your data?
- Some features can hack your target



Popularity Bias

- Popular content has more interactions in dataset

- Users prefer content with a lot of feedback(views, likes, rates)



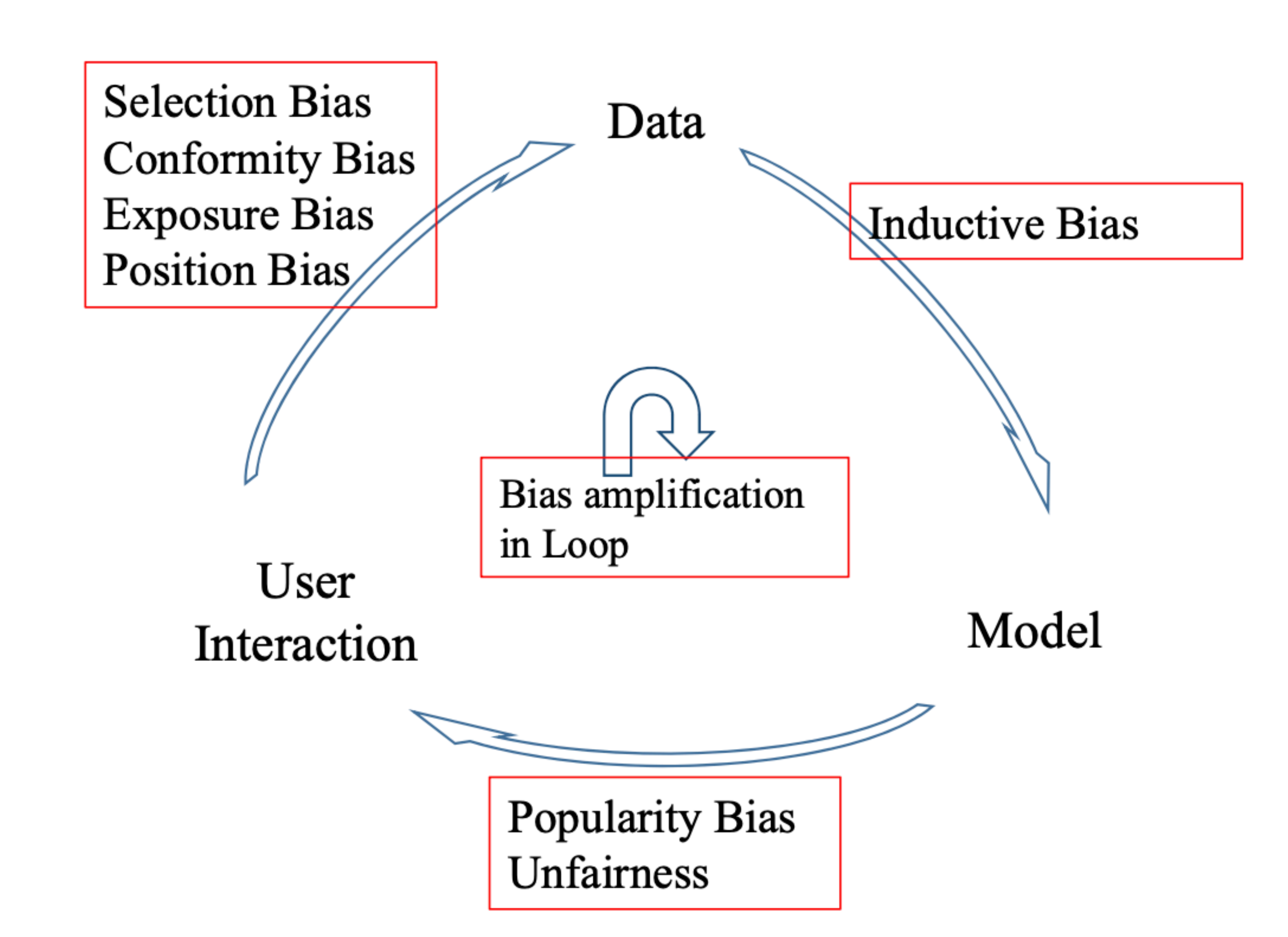
Unfairness

- Small societies has low quality recommendations
- Low number of interaction for item -> bad ranking

Feedback Loop

- The result of all biases concatenate and could be further intensified over time along the loop

- Recommendation system need to monitor this biases



Debiasing Selection Bias

Propensity Score

$$H(\hat{R}) = \frac{1}{nm}\sum_{u=1}^{n}\sum_{i=1}^{m}\delta_{u,i}(r,\hat{r})$$

$$\hat{H}_{\textbf{naive}}(\hat{r}) = \frac{1}{\left|\{(u,i) : O_{u,i}=1\}\right|}\sum_{(u,i):O_{u,i}=1}\delta_{u,i}(r,\hat{r})$$

$$\hat{H}_{IPS}(\hat{r} \mid P) = \frac{1}{nm}\sum_{(u,i):O_{u,i}=1}\frac{\delta_{u,i}(r,\hat{r})}{P_{u,i}}$$

$$\begin{aligned}\mathbb{E}_O\left[\hat{H}_{IPS}(\hat{r} \mid P)\right] &= \frac{1}{nm}\sum_u\sum_i \mathbb{E}_{O_{u,i}}\left[\frac{\delta_{u,i}(r,\hat{r})}{P_{u,i}}O_{u,i}\right]\\ &= \frac{1}{nm}\sum_u\sum_i \delta_{u,i}(r,\hat{r}) = H(\hat{r})\end{aligned}$$

Data Imputation

$$\arg\min_{\theta}\sum_{u,i} W_{u,i}\cdot\left(r_{u,i}^{o\&i} - \hat{r}_{u,i}(\theta)\right)^2 + \mathrm{Reg}(\theta)$$

Debiasing Conformity Bias

- Train XGBoost predict score without user features

$$\hat{r}_{ui} = xgb\left(\left\{(1-\omega)\cdot t_{ui} + \omega \cdot a_{ui}, c_{ui}, a_{ui}, d_{ui}\right\}, \Theta_{xgb}\right)$$

Debiasing Exposure Bias

- Update each iteration of your algorithm with different learning rate for different unobserved items
- Set for unobserved items other constant then 0
- Advanced sampling

Debiasing Position Bias

- Click modeling
- Add position to rate prediction

Debiasing Popularity Bias

- Regularization
- Adversarial learning

Debiasing Unfairness

- Define what is it
- Garante a fair chance to the Item been in recommended lists
- Rebalance your result list and garante all-group parity

Break down feedback loop

- Uniform data distribution
- Add random
- Reinforcement learning

Sources

- https://arxiv.org/pdf/2010.03240.pdf
- https://papers-gamma.link/static/memory/pdfs/153-Kunaver_Diversity_in_Recommender_Systems_2017.pdf
- https://arxiv.org/pdf/1905.06589.pdf