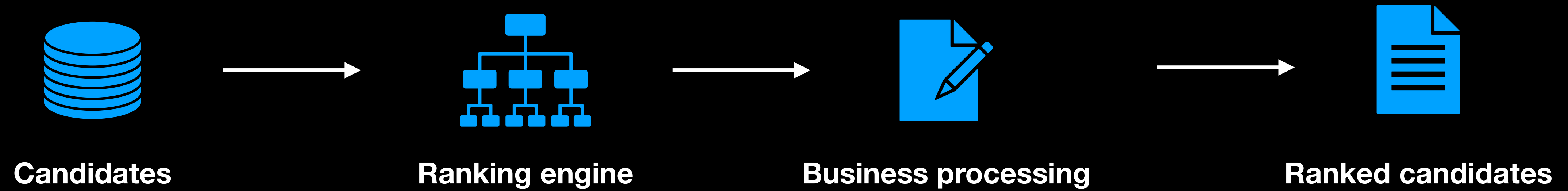


# Recommendation Systems

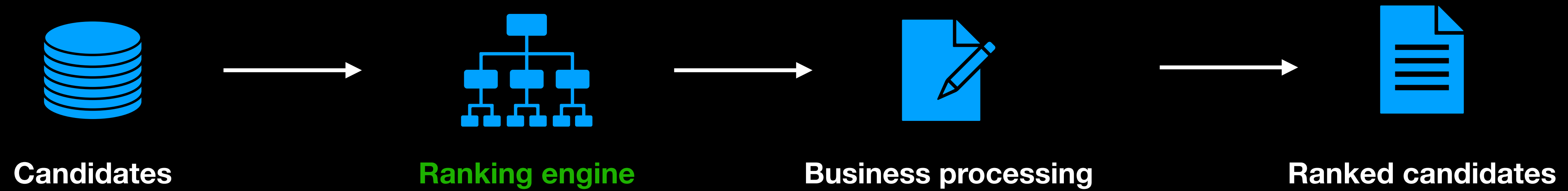
Learning to **rank**

Eugeny Malyutin / Sergey Dudorov

# RecSystem structure



# RecSystem structure



# Ok, ML

## How to define it?

- ...
- $x \in X = \{x_1, \dots, x_n\}$  — items,  $y \in Y = \{y_1, \dots, y_n\}$  — «labels»
- There are  $X_n, Y_n$  — given dataset with «answers», we believe that there is implicit dependency  $y^* : X \rightarrow Y$
- Need to define algorithm  $\alpha : X \rightarrow Y$ .
- If  $Y \in \{0,1\}$  — this task is called binary classification  
If  $Y \in \{y_0, y_1 \dots y_n\}$  — this task is called multi class classification  
If  $Y = R$  — this task is called regression

# Ok, ML


## How to define it?

- ...
- $x \in X = \{x_1, \dots, x_n\}$  — items,  $y \in Y = \{y_1, \dots, y_n\}$  — «labels»
- There are  $X_n, Y_n$  — given dataset with «answers», we believe that there is implicit dependency  $y^* : X \rightarrow Y$
- Need to define algorithm  $\alpha : X \rightarrow Y$ .
- If  $Y \in \{0,1\}$  — this task is called binary classification  
If  $Y \in \{y_0, y_1 \dots y_n\}$  — this task is called multi class classification  
If  $Y = R$  — this task is called regression

# Ok, binary classification.

## Does our cases fits binary classification?

Recommended




Wild Bags

Open group

2 friends · 48,827 members

Follow




Пикабу

Humor

58 friends · 3,057,449 followers

Follow




Cossa

Mass media

12 friends · 165,595 followers

Follow




Ton Twitter

Humor

8 friends · 99,993 followers

Follow




PYE

Open group

6 friends · 42,068 members

Follow




Semrush

Websites

4 friends · 2,531 followers

Follow



Albertina


Gallery

19 friends · 950,042 followers

Follow

Возможно, вы знакомы

Show all >




Vladislav Kaverin

ВКонтакте

29 общих друзей

+ Add




Natasha Belay

ВКонтакте

36 общих друзей

+ Add




Ksenia Timofeeva

Пермь

4 общих друга

+ Add


Customers who viewed this item also viewed



Daily Ritual Women's Lived-in Cotton Short-Sleeve Crewneck Maxi Dress

★★★★☆ 42


\$23.40 - \$26.00



Daily Ritual Women's Jersey Mock-Neck Maxi Dress

★★★★☆ 34


\$24.50



Daily Ritual Women's Jersey Sleeveless V-Neck Dress

★★★★☆ 154


\$20.00



Daily Ritual Women's Jersey Crewneck Muscle Sleeve Maxi Dress with Side Slit

★★★★☆ 43


\$20.40 - \$24.50



ZYX Women You are My Sunshine Letter Print Tops Casual Short Sleeve Tee Rainbow T-Shirt

★★★★☆ 1


\$16.98



Daily Ritual Women's Supersoft Terry Hooded Short-Sleeve Sweatshirt

★★★★☆ 16

\$25.20 - \$28.00



Amazon Essentials Women's Tank Maxi Dress

★★★★☆ 24

\$26.00

# Learning to **rank**

## Definition

- $X$  — set of objects. Exists:  $x_i \prec x_j$  — **order** relation.

Our task to implement ranking function:  $x_i \prec x_j \rightarrow a(x_i) < a(x_j)$

- Most of times order exists alongside with groupId (query relevant documents search engine)  
if  $y(q, d) = \{0, 1\}$  — binary relevance,  $d_q^{(i)}$  - i-th doc by  $a(x_i)$  desc
- Ok, what's do you need more?
- Metrics
- Algorithms
- Tricks

# Learning to **rank**

## Metrics

- **Precision:**  $P_n(q) = 1/n \sum_{i=1}^n y(q, d_q^{(i)})$  (percentage of correct in first n).
- Problems?
- **Average Precision:**  $AP(q) = \sum y(d, d_q^{(n)})P_n(q) / \sum y(q, d_n^q)$   
(average  $P_n$  by all relevant docs position)
- Problems?
- **Mean average precision:**  $MAP = 1/|Q| \sum_q AP(q)$



# Learning to **rank**

## Metrics

- **MRR:**  $MRR = 1/|Q| \sum_{i=1}^{|Q|} 1/rank_i$ 
  - Sum over relevant!
  - Easy to interpret
- **DCG** (discounted cumulative gain):  $DCG_n(q) = \sum_i^n G_q(d_q^i) D(i)$ 

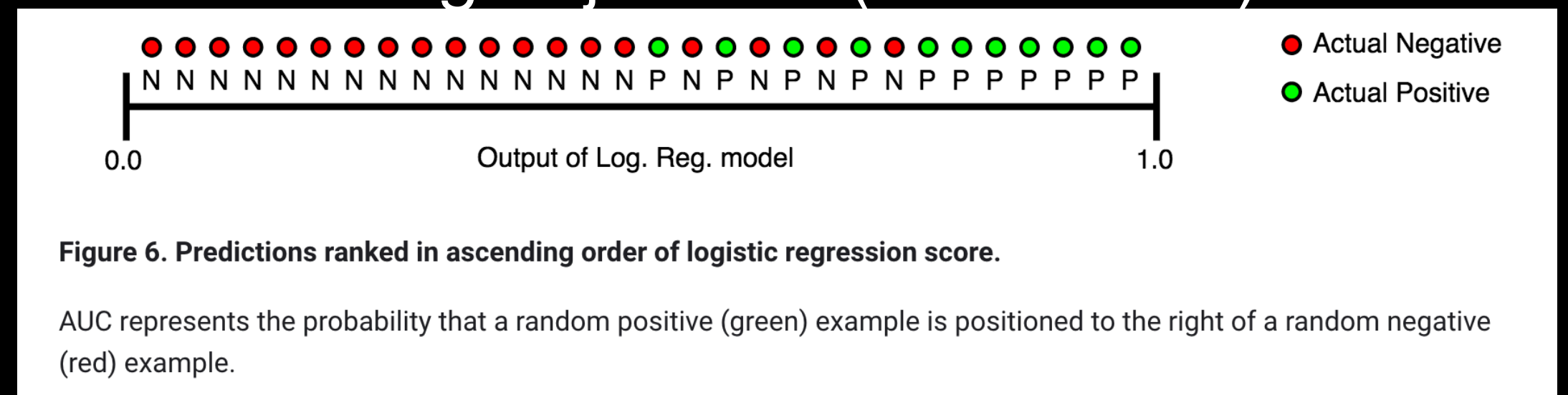
$G_q(d_q^i) = (y(d, q))$  — gain, 1/0 for relevant/unrelevant doc's

$D(i) = 1/\log_2(i + 1)$  — discounts, higher relevant docs are — better
- **NDCG:**  $NDCG(q) = DCG_n(q)/maxDCG_n(q)$  — normalized version

# Learning to **rank**

## Metrics

- **DCG** (discounted cumulative gain):  $DCG_n(q) = \sum_i^n G_q(d_q^i) D(i)$   
 $G_q(d_q^i) = (y(d, q))$  — gain, 1/0 for relevant/unrelevant doc's  
 $D(i) = 1/\log_2(i + 1)$  — discounts, higher relevant docs are — better
- **NDCG**:  $NDCG(q) = DCG_n(q) / \max DCG_n(q)$  — normalized version
- Also NDCG exists w.r.t. groupId, possible to use averaged over groups.
- Strongly correlates with AUC-per-groupId, but works with ranking objectives (unbounded)



# Learning to rank

## NDCG by example

### NDCG - Example

4 documents:  $d_1, d_2, d_3, d_4$

i	Ground Truth		Ranking Function <sub>1</sub>		Ranking Function <sub>2</sub>	
	Document Order	r <sub>i</sub>	Document Order	r <sub>i</sub>	Document Order	r <sub>i</sub>
1	d4	2	d3	2	d3	2
2	d3	2	d4	2	d2	1
3	d2	1	d2	1	d4	2
4	d1	0	d1	0	d1	0
	NDCG <sub>GT</sub> =1.00		NDCG <sub>RF1</sub> =1.00		NDCG <sub>RF2</sub> =0.9203	

$$DCG_{GT} = 2 + \left( \frac{2}{\log_2 2} + \frac{1}{\log_2 3} + \frac{0}{\log_2 4} \right) = 4.6309$$

$$DCG_{RF1} = 2 + \left( \frac{2}{\log_2 2} + \frac{1}{\log_2 3} + \frac{0}{\log_2 4} \right) = 4.6309$$

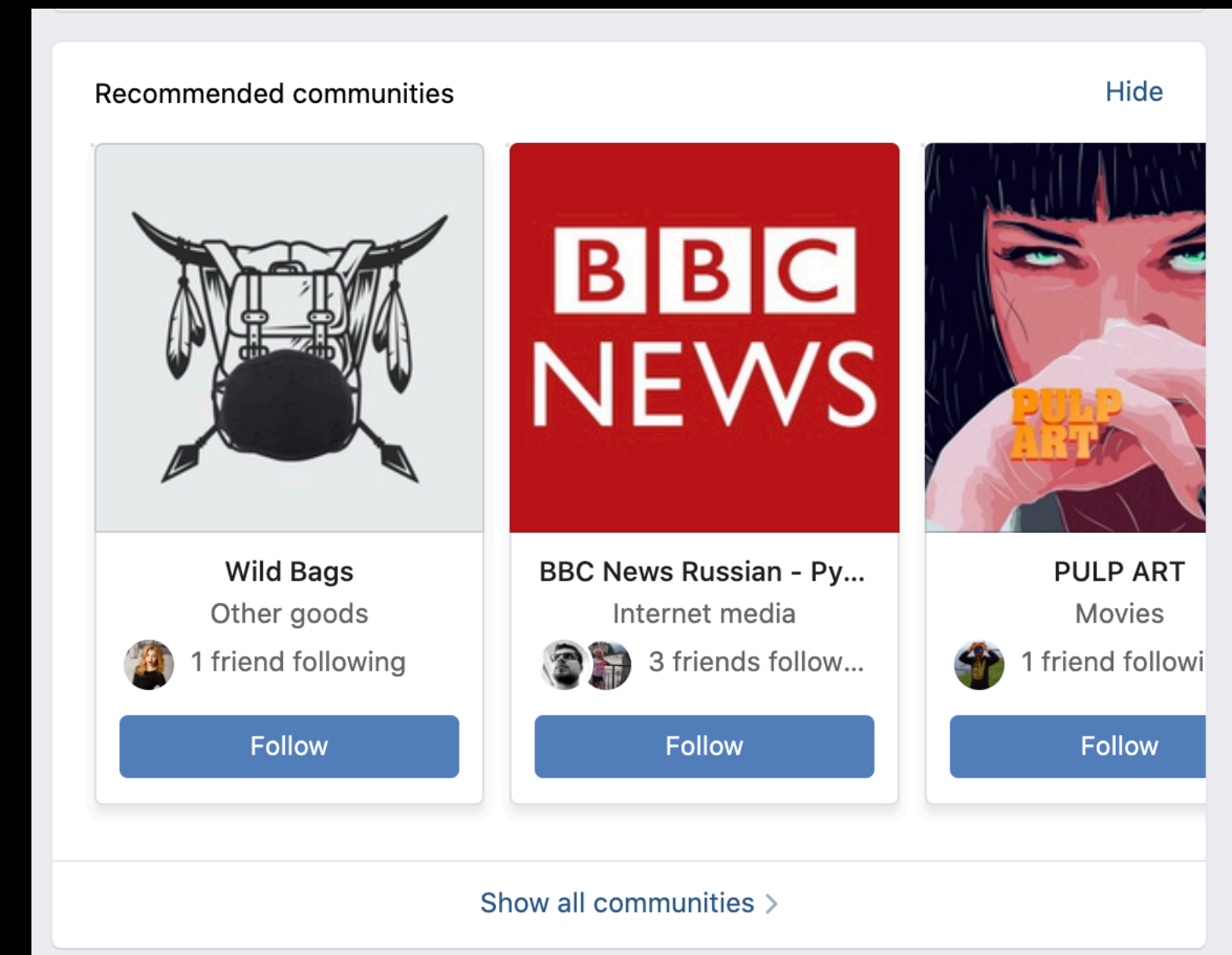
$$DCG_{RF2} = 2 + \left( \frac{1}{\log_2 2} + \frac{2}{\log_2 3} + \frac{0}{\log_2 4} \right) = 4.2619$$

$$MaxDCG = DCG_{GT} = 4.6309$$

# Learning to **rank**

## Metrics

- **DCG** (discounted cumulative gain):  $DCG_n(q) = \sum_i^n G_q(d_q^i) D(i)$   
 $G_q(d_q^i) = (y(d, q))$  — gain, 1/0 for relevant/unrelevant doc's  
 $D(i) = 1/\log_2(i + 1)$  — discounts, higher relevant docs are — better
- **NDCG**:  $NDCG(q) = DCG_n(q) / \max DCG_n(q)$  — normalized version
- Also popular **NDCG-at-k** (k=3, for example)



# Learning to **rank**

## Classification ontology

- **Pointwise:** sorting by score  $\alpha(x_i)$  estimating «relevance»
- **Pairwise:** compares objects  $a(x_i, x_j) :> 0$  if  $x_i < x_j$  — decide which one is more relevant
- **Listwise:** Takes the entire list of candidates and optimise it's order



# Learning to **rank**

## Classification ontology

- **Pointwise:** sorting by score  $\alpha(x_i)$  estimating «relevance»

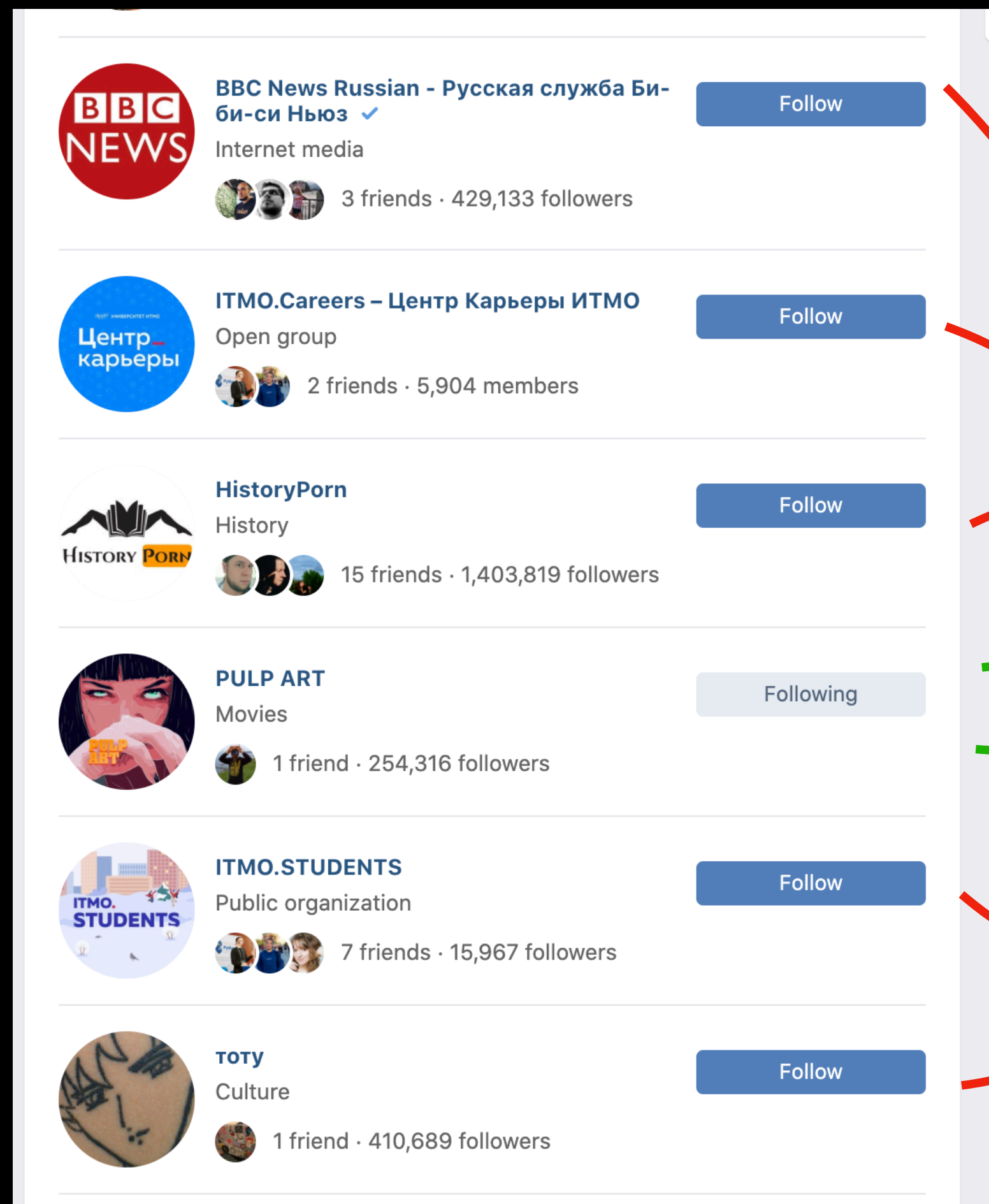
Profile	Category	Followers	Relationship	Classification
BBC News Russian - Русская служба Би-би-си Ньюз	Internet media	3 friends · 429,133 followers	Follow	Negative
ИТМО.Careers – Центр Карьеры ИТМО	Open group	2 friends · 5,904 members	Follow	Negative
HistoryPorn	History	15 friends · 1,403,819 followers	Follow	Negative
PULP ART	Movies	1 friend · 254,316 followers	Following	Positive
ИТМО.STUDENTS	Public organization	7 friends · 15,967 followers	Follow	Negative
тоту	Culture	1 friend · 410,689 followers	Follow	Negative

- **Positives** —  $y_3 = 1$
- **Negatives** —  $y_{0,1,2,4,5} = 0$
- $\alpha(x)$  gives relevance score  $\in [0,1]$

# Learning to **rank**

## Classification ontology

- **Pairwise:** decide which one is more relevant



**Negative**

**Negative**

**Negative**

**Positive**

**Negative**

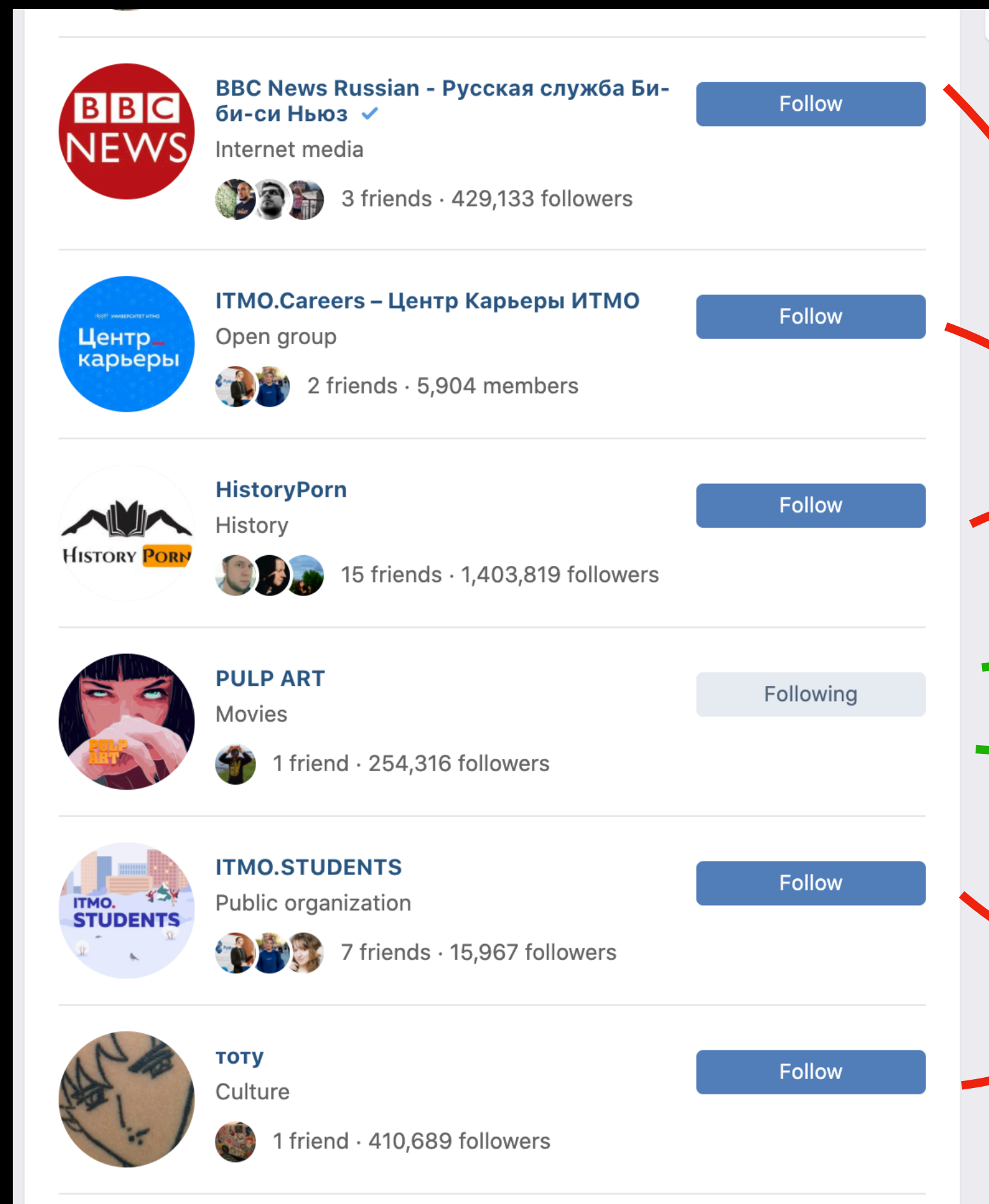
**Negative**

- **Train set:**  $\{x_3 \prec x_0, x_3 \prec x_1, \dots, x_3 \prec x_5\}$
- $\alpha(x_i, x_j)$  compares pair of objects
- OK, what's a problem?

# Learning to **rank**

## Classification ontology

- **Pairwise:** decide which one is more relevant



**Negative**

**Negative**

**Negative**

**Positive**

**Negative**

**Negative**

- **Train set:**  $\{x_3 \prec x_0, x_3 \prec x_1, \dots, x_3 \prec x_5\}$

- $\alpha(x_i, x_j)$  compares pair of objects

- OK, what's a problem?

- How much  $\alpha$  applications do you need to sort 10000 - elements list?



# Learning to **rank**

## Classification ontology

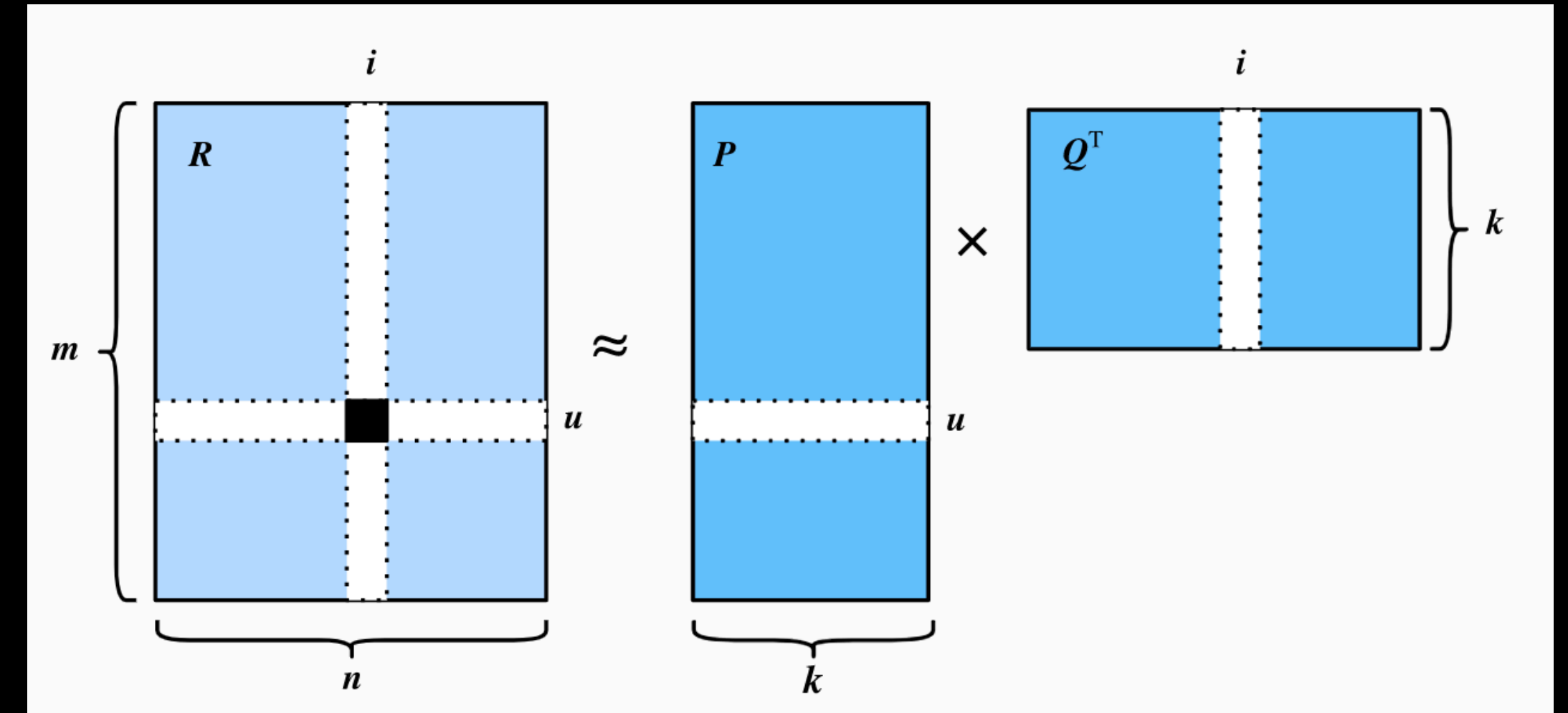
- **Listwise:** optimize the whole list



# Learning to **rank**

## SVD-factorisation

$$\operatorname{argmin}_{P,Q,b} \sum_{u,i \in K} ||R_{ui} - \hat{R}_{ui}||^2 + \lambda(||P||^2 + ||Q||^2 + b_u^2 + b_i^2)$$



$$\operatorname{argmin}_{p,q,b} \sum_{u,i \in K} ( \langle p_u, q_i \rangle + b_u + b_i + \mu - x_{ij} )^2 + \alpha \sum_i ||u_i||^2 + \beta \sum_j ||p_j||^2 + \gamma ||b_u|| + \theta ||b_i||$$

$$\alpha(u, i) = \langle p_u, q_i \rangle + b_u + b_i + \mu \longrightarrow \text{Pointwise algorithm(!)}$$

# Learning to **rank**

## BPR-factorisation

$$p(\Theta \mid >_u) \propto p(>_u \mid \Theta)p(\Theta)$$



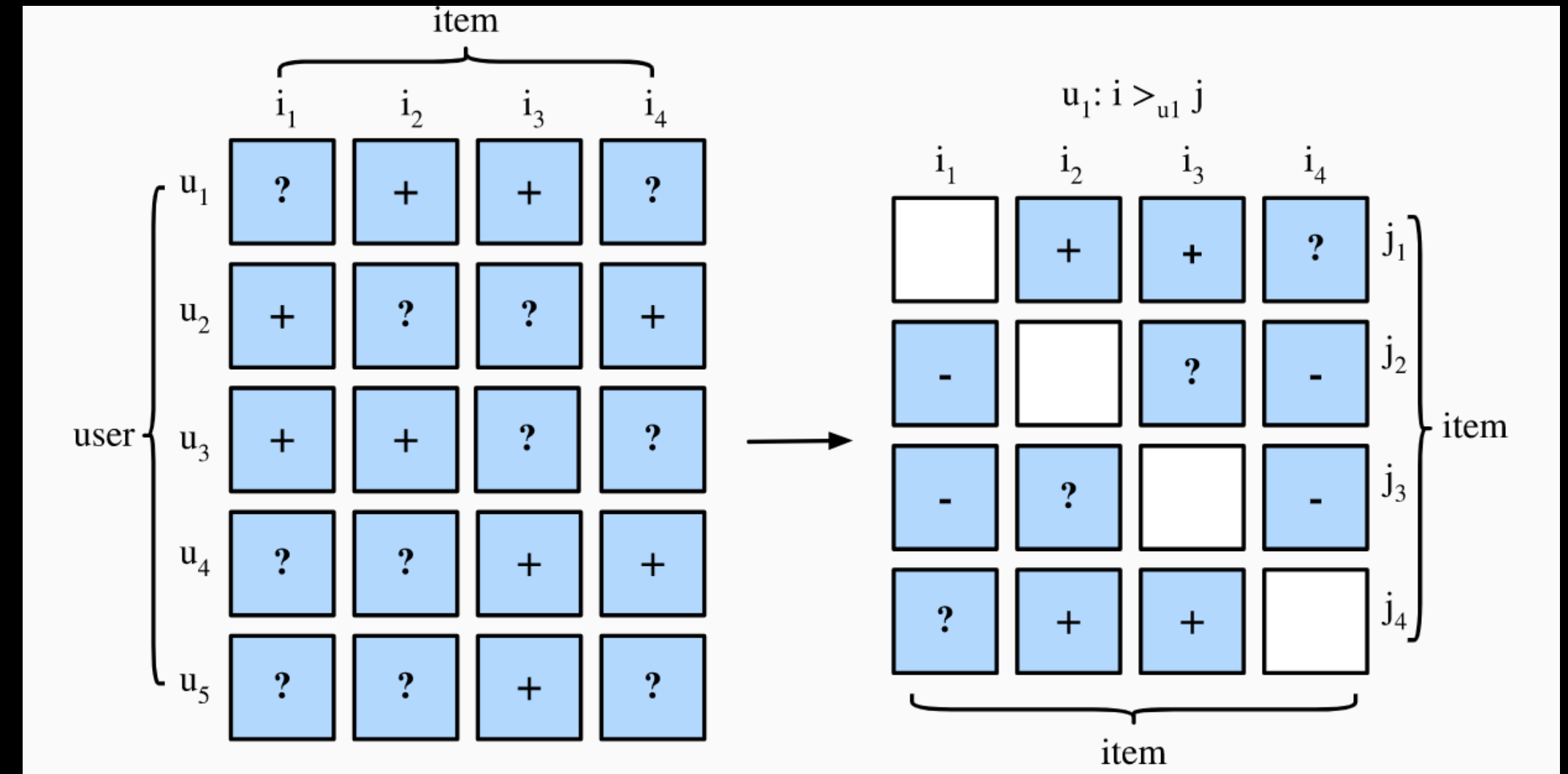
$$\text{BPR-OPT} := \ln p(\Theta \mid >_u)$$

$$\propto \ln p(>_u \mid \Theta)p(\Theta)$$

$$= \ln \prod_{(u,i,j \in D)} \sigma(\hat{y}_{ui} - \hat{y}_{uj})p(\Theta)$$

$$= \sum_{(u,i,j \in D)} \ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \ln p(\Theta)$$

$$= \sum_{(u,i,j \in D)} \ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) - \lambda_{\Theta} \|\Theta\|^2$$



$$r_{ui} = \langle e_u, e_i \rangle + b_u + b_i + \mu$$

$$\alpha(x_i, x_j, u) = \ln(\sigma(\langle e_u, e_i \rangle + b_u + b_i - \langle e_u, e_j \rangle - b_u - b_j))$$



$$\alpha(u, i) = \langle e_u, e_i \rangle$$

**Sort** with  $\alpha$

**PAIR**wise algorithm while learn  
**POINT**wise algorithm on runtime!

# Learning to **rank**

## Summary

- Ranking **differs** either classification or regression due to strict **order properties** on **relevance**
- It leads us to new metrics: MAP, MRR, AUC(~), NDCG
- Algorithms ontology: **point-**, **pair-**, **list-wise**
- There are **no** working «fair» **listwise** algorithms **IRL** (at least author saw no one)
- **Best pairwise** algorithms — camouflages themselves as **pointwise**.