Toloka

Toloka

WWW 2021 hands-on tutorial

Improving Web Ranking with Human-in-the-Loop: Methodology, Scalability, Evaluation

Alexey Drutsa, Dmitry Ustalov, Nikita Popov, Daria Baidakova

Part V
Pairwise Comparisons

Dmitry Ustalov,
Analyst/Software Developer

Tutorial schedule

Introduction:
45 min

Part I: 60 min
Ranking and Quality Metrics

Part II: 45 min
Human-in-the-Loop Essentials

Part III: 90 min
Practice Session I

Lunch Break:
60 min

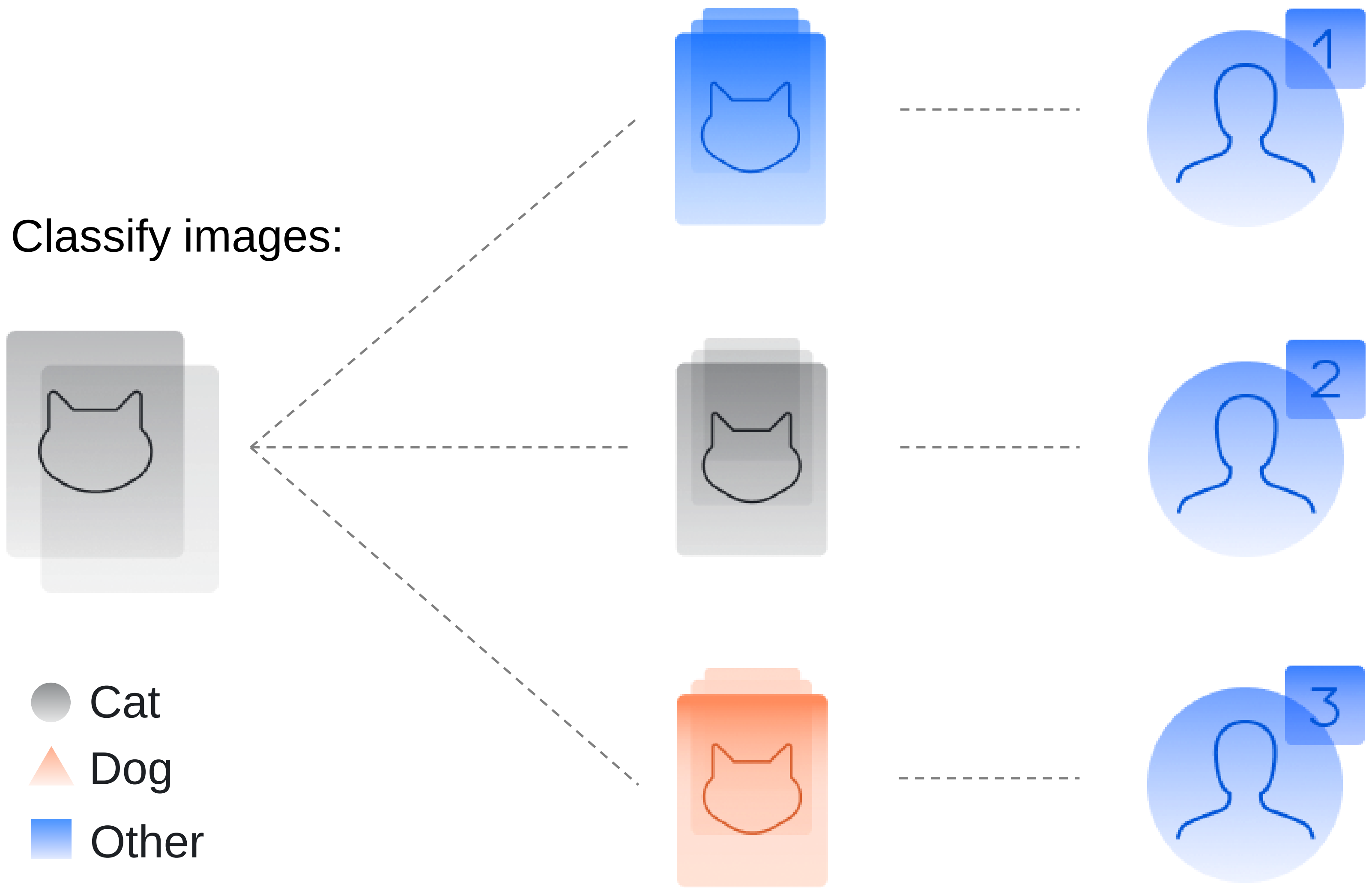
Part IV: 30 min
Practice Session II

Part V: 60 min
Pairwise Comparisons

Part VI: 30 min
Final Remarks and Conclusion

Labelling Data with Crowdsourcing

Classify images:



- How to choose a reliable label?
- How many performers per object?
- How much to pay to performers?
- …

Evaluation of Labelling Approaches

vs

Accuracy

Cost

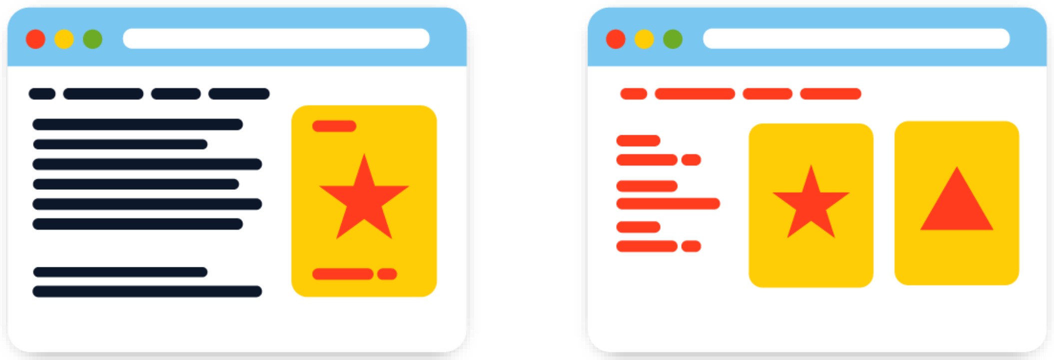
- Labels with a maximal level of accuracy for a **given budget**

or

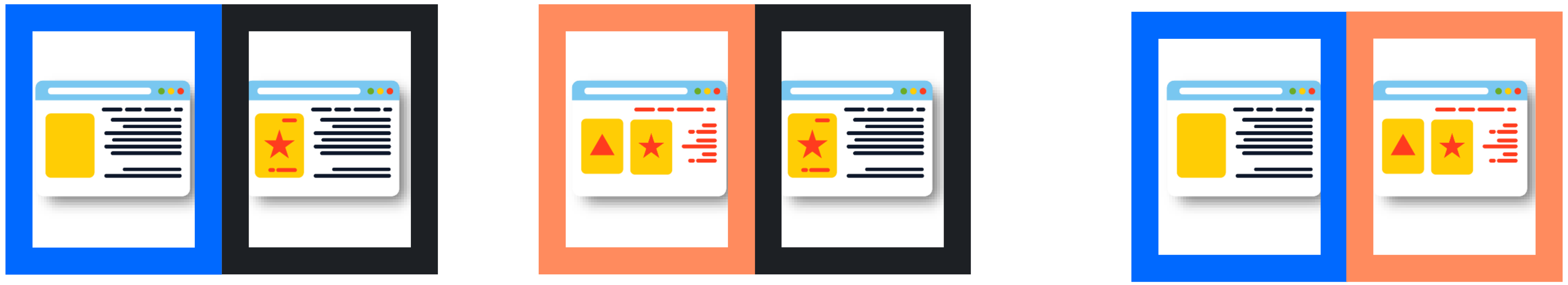
- Labels of a chosen accuracy level for a **minimal budget**

Difference from Multiclassification

- The latent label assumption is not satisfied when comparing complex items

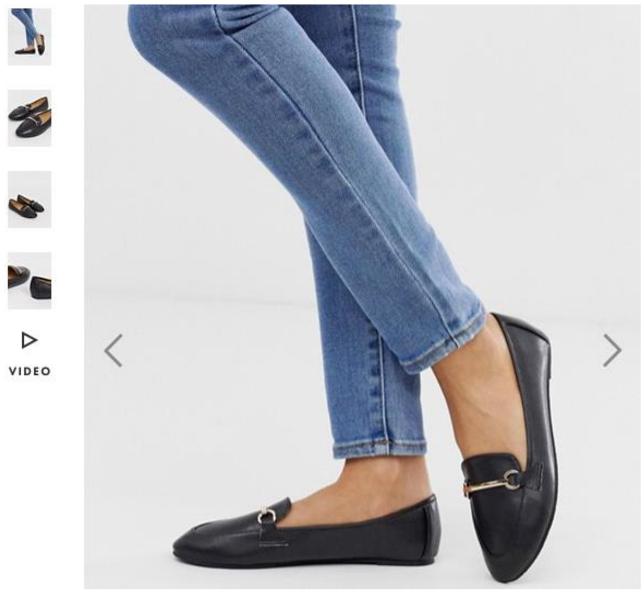


- Different tasks may contain common items



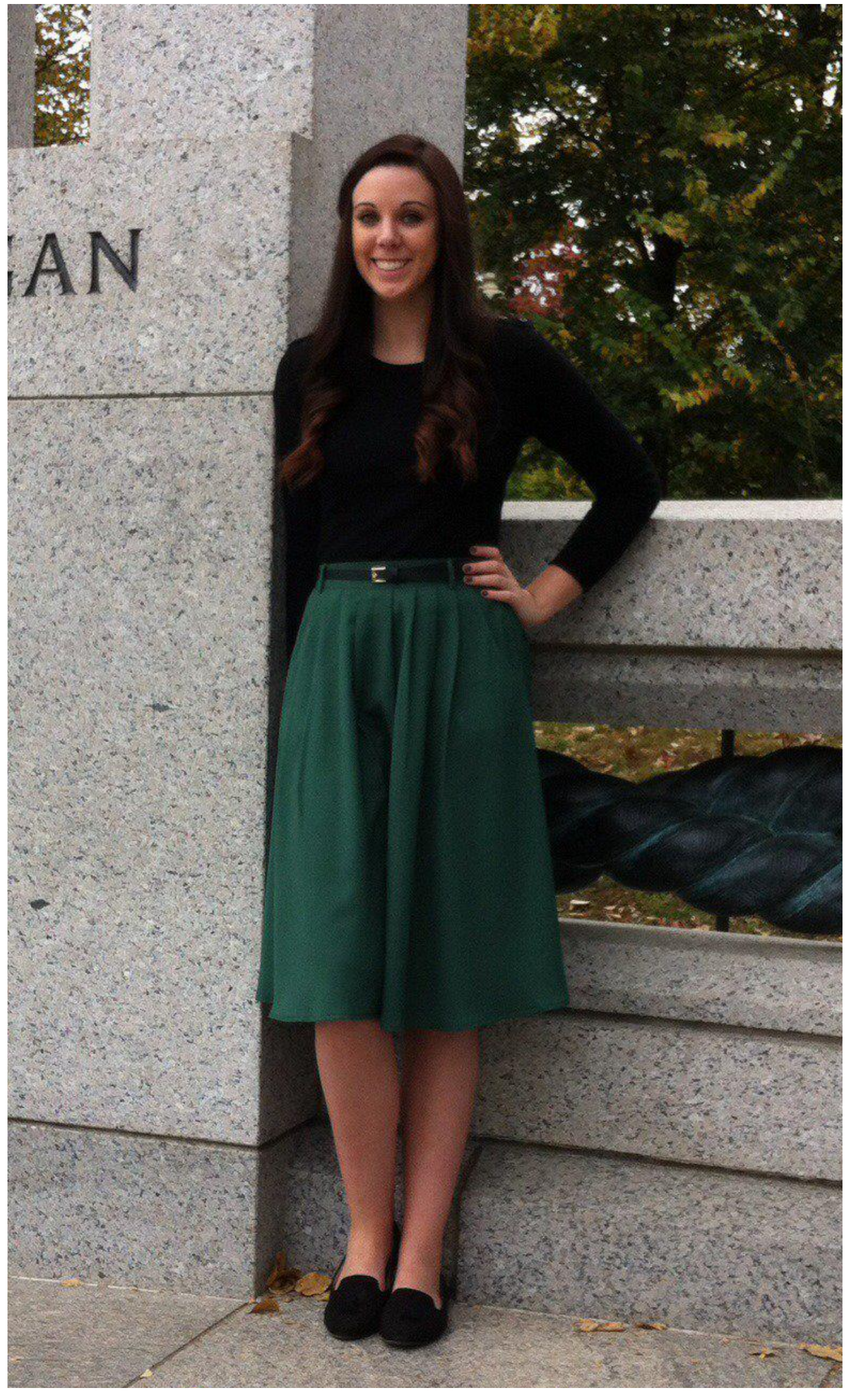
Task: Compare Items

Which shoes look more similar to the one in the picture?



Left

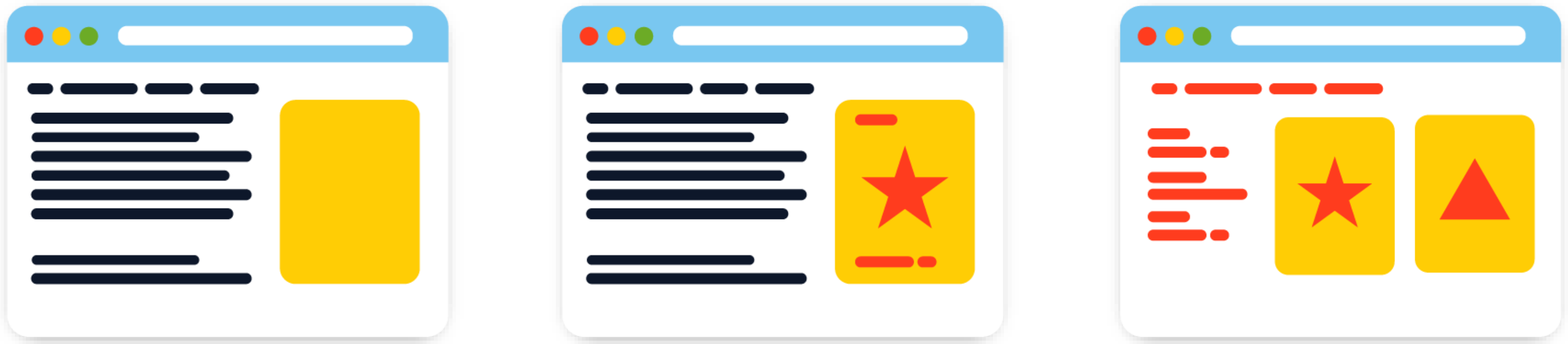
Right



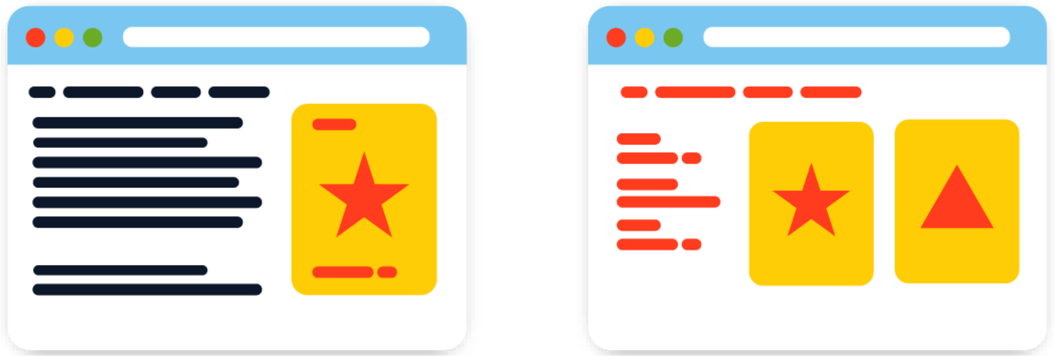
Aggregating Pairwise Comparisons

Notation

- Answers: Left or Right
- Items $d_j \in \{1, \dots, N\}$ E.g.:

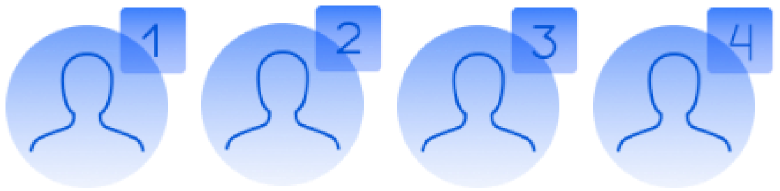


- Tasks:



Choose a better item:
Left
Right

- Performers $w \in \{1, \dots, W\}$ E.g.:



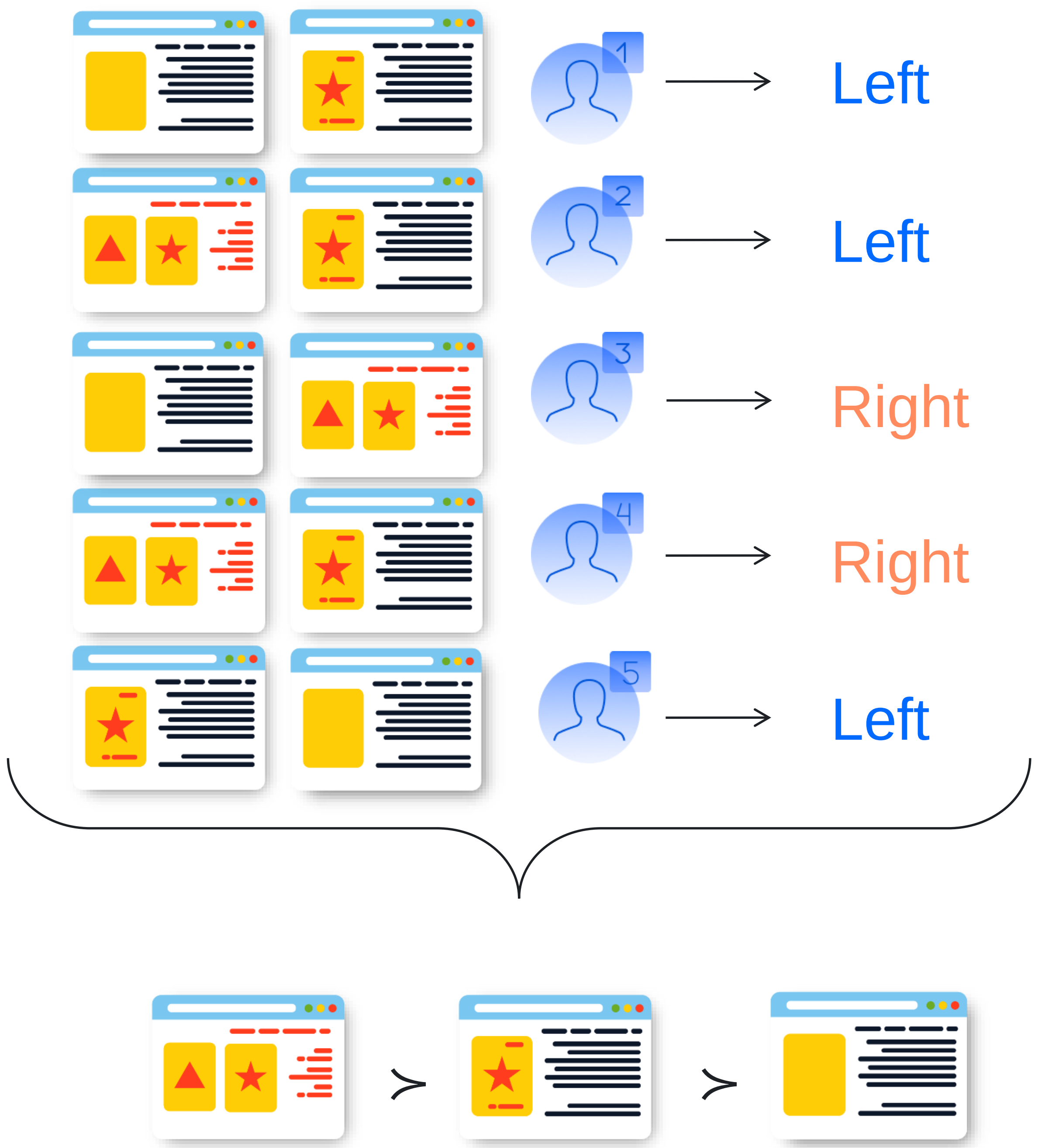
Formalization

Ranking from pairwise comparisons:

- Given pairwise comparisons for items in D:

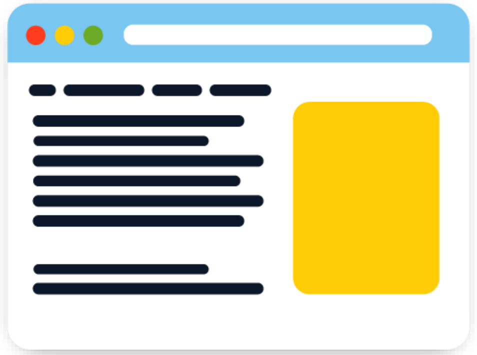
$$P = \{(w_k, d_i, d_j) : i \succ_k j\}$$

- Obtain **a ranking** π over items $D \to \{1, \ldots, N\}$ based on answers in P



Bradley and Terry Model (BT)

- Assume that each item $d_i \in D$ has a latent "quality" score $s_i \in \mathbb{R}$

d_i  $\longleftrightarrow$ s_i

- The probability that $d_i \in D$ will be preferred in a comparison over $d_j \in D$

$$\Pr(i \succ j) = f\left(s_i - s_j\right), \text{ where } f(x) = {}^{1}/_{1+e^{-x}}.$$

Bradley, R. A. and Terry, M. E. "Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons". 1952

Bradley and Terry Model: Example

Performer	Task	Left	Right
w_1	t_1	**a**	b
w_1	t_2	**b**	c
w_1	t_3	c	**a**
w_2	t_1	**a**	b
w_2	t_2	**b**	c
w_2	t_3	**c**	a

Item	Score
a	0.592
b	0.278
c	0.130

The model assumes that all performers are equally good and truthful!

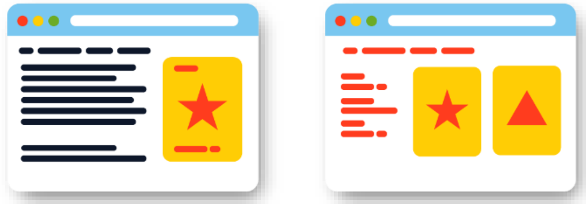
NoisyBT Model: Parameterization of Performers

w_k  $\longleftrightarrow$ "reliability" γ_k and "bias" q_k

- The probability that w reads task is

$$\Pr(w_k \text{ reads a task}) = \mathrm{f}(\gamma_k) \longleftarrow \text{Logistic function}$$

- If w_k reads a task, she answers according to scores:



$\updownarrow$ $\mathrm{s_j}$ $\qquad$ $\updownarrow$ $\mathrm{s_i}$

$$\underbrace{(\mathrm{f}(\mathrm{s_i} - \mathrm{s_j})}_{\text{Probability to choose Left if compares items}}, \mathrm{f}(\mathrm{s_j} - \mathrm{s_i}))$$

- If w_k does not read a task, she answers according to her bias

$$\underbrace{\mathrm{f}(q_k)}_{\text{Probability to choose Left if answers randomly}}, \mathrm{f}(-q_k)$$

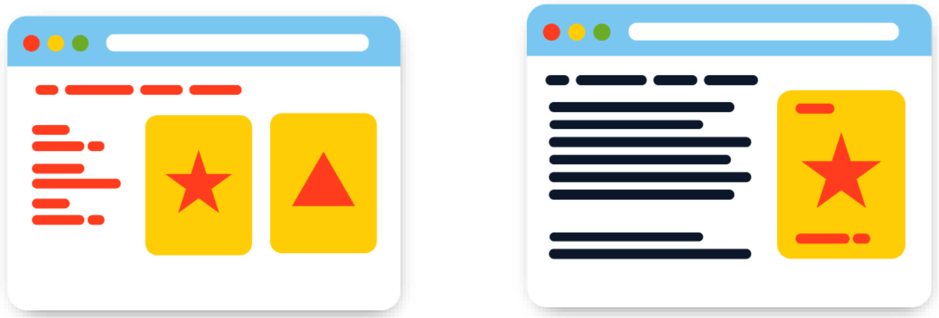
NoisyBT: Answer Likelihood

The likelihood of $i \succ_k j$ is

$$\Pr(i \succ_k j) = \underbrace{f(\gamma_k) f\left(s_i - s_j\right)}_{\text{Truthful answer}} + \underbrace{\left(1 - f(\gamma_k)\right) f\left((-1)^{(1 - \mathbb{I}(d_i \text{ was left}))} q_k\right)}_{\text{Random answer}},$$

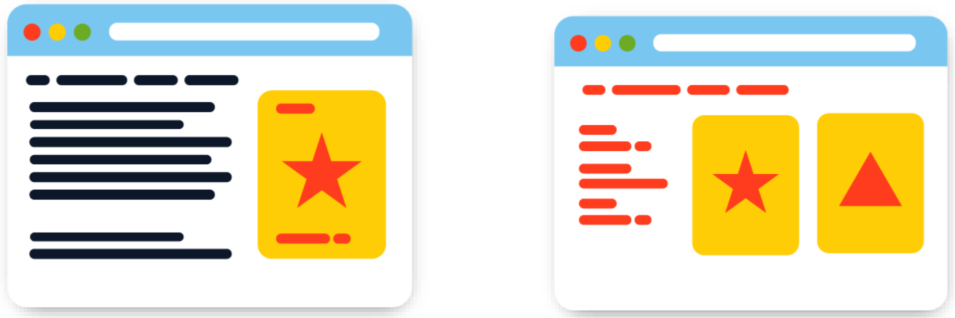
where $\mathbb{I}(d_i \text{ was left})$ is the indicator for the order of d_i and d_j

d_i d_j



$\mathbb{I}(d_i \text{ was left}) = 1$

d_j d_i



$\mathbb{I}(d_i \text{ was left}) = 0$

NoisyBT: Parameter Estimation

Likelihood of observed comparisons:

$$T(s, q, \gamma) = \sum_{(w_k, d_i, d_j) \in P} \log \Pr(i \succ_k j) =$$

$$\sum_{(w_k, d_i, d_j) \in P} \log\left[f(\gamma_k) f\left(s_i - s_j\right) + \left(1 - f(\gamma_k)\right) f\left((-1)^{(1 - \mathbb{I}(d_i \text{ was left}))} q_k\right)\right]$$

- $\{s_i\}_{i=1,\ldots,N}$ and $\{\gamma_k, q_k\}_{k=1,\ldots,W}$ are inferred by maximizing the log-likelihood:

$$T(s, q, \gamma) \to \max_{\{s_i, \gamma_k, q_k\}}$$

- To obtain a **ranking** π over items, **sort** items according to their **scores**

NoisyBT: Example

Performer	Task	Left	Right
w_1	t_1	**a**	b
w_1	t_2	**b**	c
w_1	t_3	c	**a**
w_2	t_1	**a**	b
w_2	t_2	**b**	c
w_2	t_3	**c**	a

Item	Score
a	1.000
b	0.547
c	0.000

Performer	Bias	Skill
w_1	0.633	0.656
w_1	1.000	0.000

Summary about NoisyBT

- Latent scores models for ranking from pairwise comparisons:

$$s_A \geq s_B \geq s_C$$

- To reduce bias from unreliable answers parameterize workers

w_k  $\longleftrightarrow$ “reliability” γ_k and “bias” q_k

Demo

Demo

- We will learn how to aggregate your results using the Bradley-Terry model **right now**
- We will show you a **live demo**
- This demo will use the annotated **data from the practice session** that will be aggregated to provide the final rankings
- Please use a blank **Jupyter Notebook**, e.g., https://colab.research.google.com/ or the local one

Crowd-Kit: https://github.com/Toloka/crowd-kit

Crowd-Kit is an open source **Python** library that implements methods for quality control in crowdsourcing.

- It is platform-agnostic, so can be used with any crowdsourcing platform
- It includes efficient implementations of classic and state-of-the-art methods for quality control
- It provides a simple API for using them in your application and is available on **PyPI**: `pip install crowd-kit`

NoisyBT with Crowd-Kit: Input

```
import pandas as pd # pip install pandas
from crowdkit.aggregation import NoisyBradleyTerry # pip install -U crowd-kit

# In this example we will use the annotation results in the Toloka TSV format,
# but Crowd-Kit is platform-agnostic and it can handle any other format
df = pd.read_csv('assignments.tsv', sep='\t', dtype=str)
df.drop(df[~df['OUTPUT:result'].isin({'L', 'R'})].index, inplace=True)

# We need to reorganize our data frame to contain the following columns:
# query, performer, left, right, label
```

NoisyBT with Crowd-Kit: Rename

```
df['performer'] = df['ASSIGNMENT:worker_id’]

df['left'] = list(zip(df['INPUT:query'], df['INPUT:link_left']))
df['right'] = list(zip(df['INPUT:query'], df['INPUT:link_right']))

df.loc[df['OUTPUT:result'] == 'L', 'label'] = \
       df.loc[df['OUTPUT:result'] == 'L', 'left’]

df.loc[df['OUTPUT:result'] == 'R', 'label'] = \
       df.loc[df['OUTPUT:result'] == 'R', 'right']
```

NoisyBT with Crowd-Kit: Apply and Output

```
bt = NoisyBradleyTerry(n_iter=5000)
result = bt.fit_predict(df)

index = pd.MultiIndex.from_tuples(result.index)
df_result = pd.DataFrame(result, columns=['score'], index=index)
df_result['query'] = result.index.str[0]
df_result['url'] = result.index.str[1]
df_result.reset_index(drop=True, inplace=True)
df_result.sort_values(['query', 'score'], ascending=[True, False],
inplace=True)
df_result[['query', 'url', 'score']]
df_result.to_csv('aggregated.tsv', sep='\t', index=False,
              columns=['query', 'url', 'score'])
```

Get Ready!

Conclusion

Conclusion

- Human-in-the-Loop improves Web ranking by gathering **high-quality pairwise comparisons**
- Aggregated rankings can be used for evaluating your service against data from the **real users**
- **Crowd-Kit** allows aggregating human judgements in a simple way: https://github.com/Toloka/crowd-kit or `pip install crowd-kit`

Thank you!
Questions?

Dmitry Ustalov

Analyst/Software Developer

dustalov@yandex-team.ru

https://research.yandex.com/tutorials/crowd/www-2021