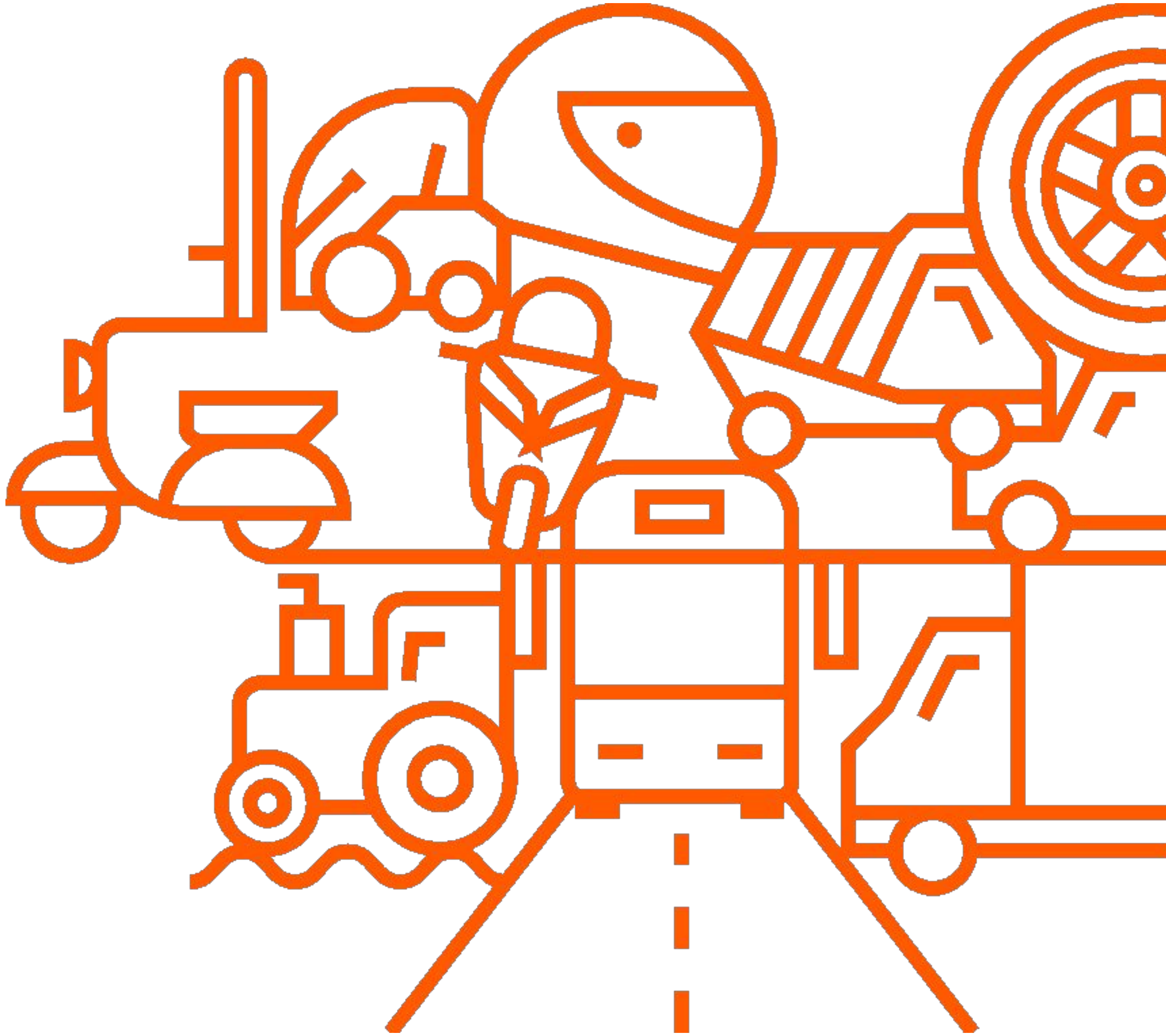
Learning to rank @ allegro.pl

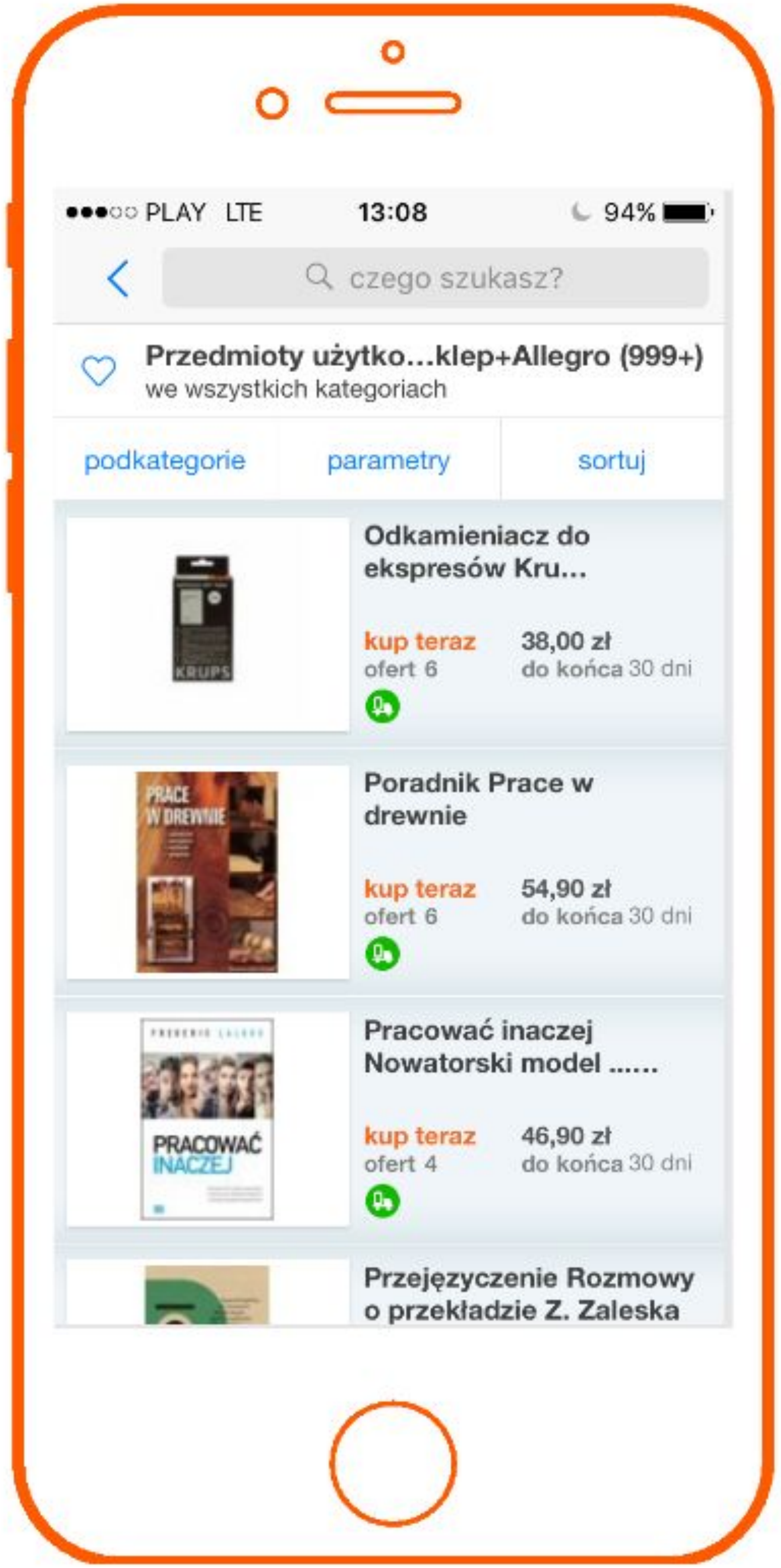
PyData Warsaw 2018

Tomasz Bartczak
Ireneusz Gawlik

allegro

Allegro - about us





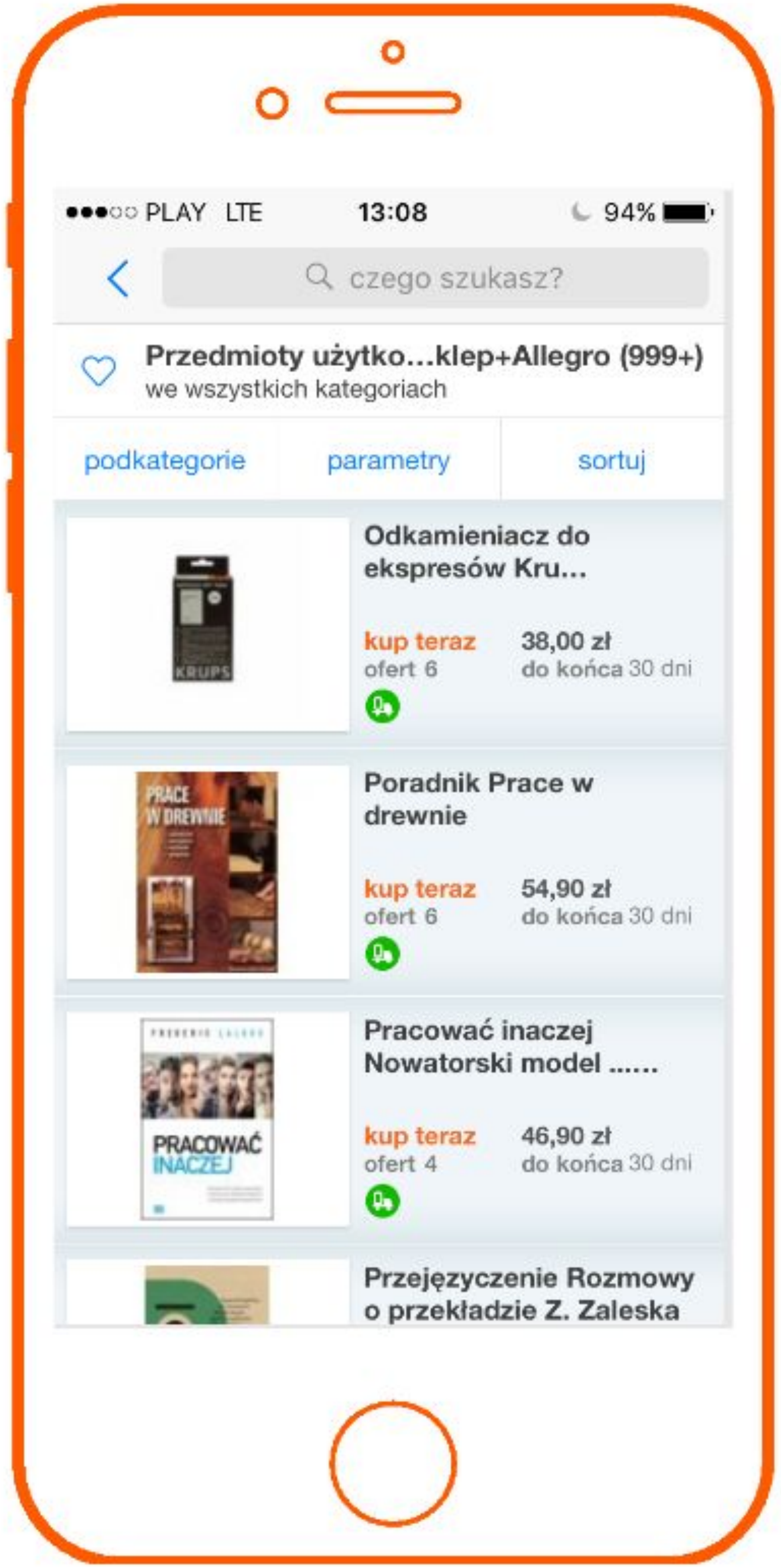
Ranking in large scale search engines

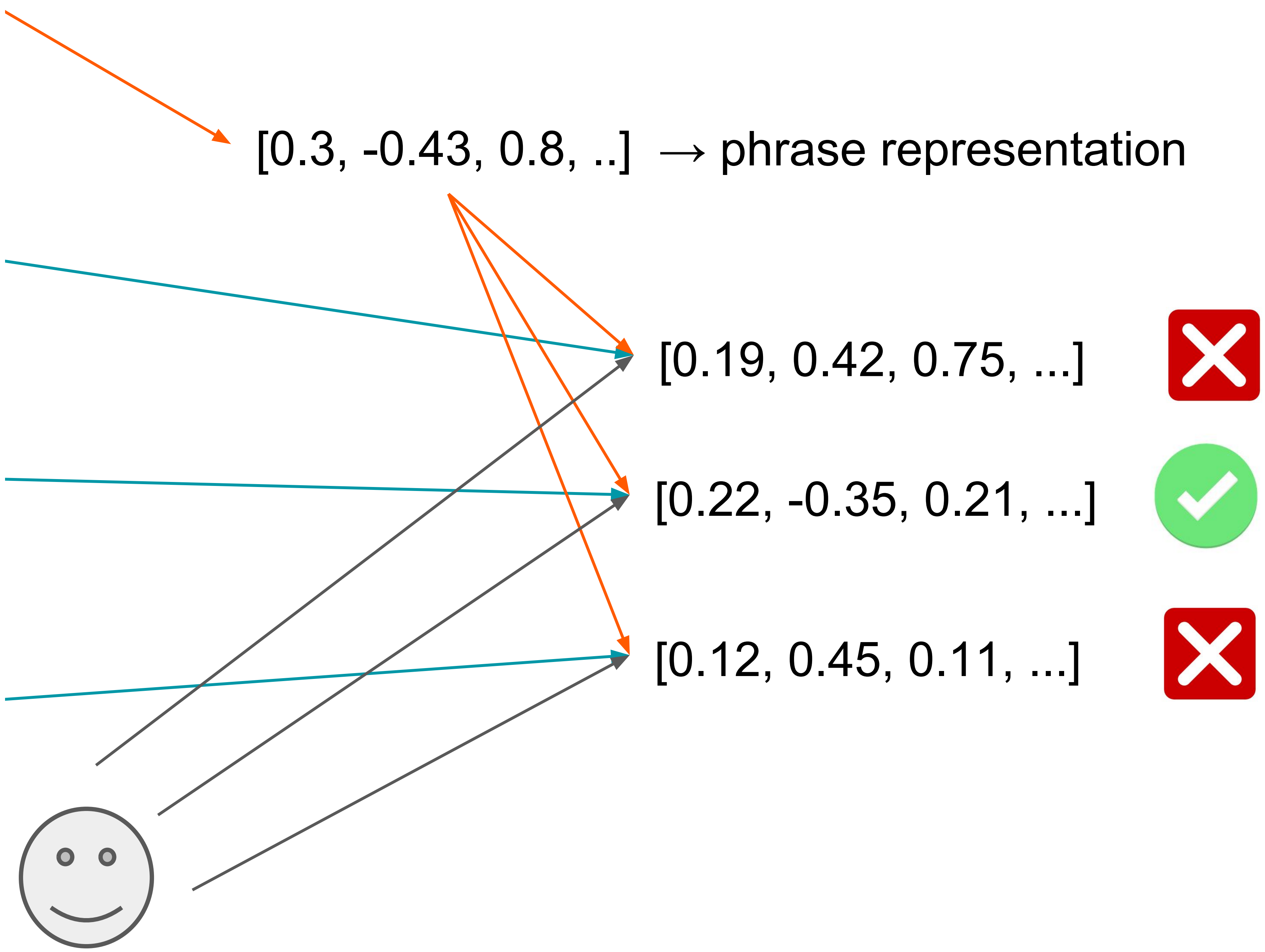
- Ranking engines are a crucial part of any search based solutions - key part of Allegro.pl.
- Ranking functions used to be hand-crafted by domain experts.
- Personalized and more accurate search results need machine learning.
- Machine learning for ranking has a coined term: *Learning to rank*.

Learning to rank

- Learning to Rank – creating a model that ranks documents for a query.
- Actual relevance scores are expensive.
- In practice, we use implicit feedback provided by users (clicks or transactions).

Learning to Rank



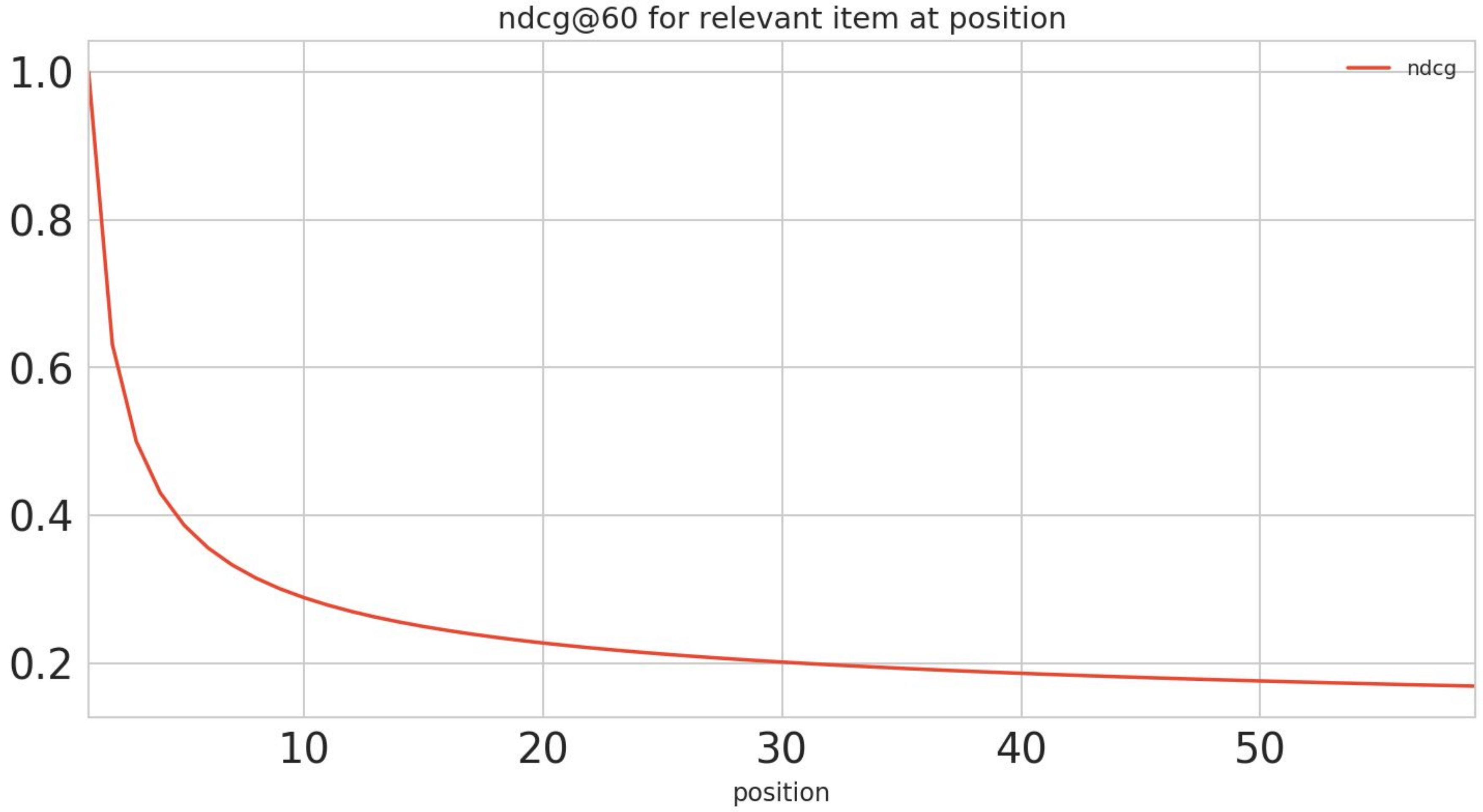


Metric: NDCG

A typical metric for ranking is NDCG[1] (Normalized Discounted Cumulative Gain)

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)}$$

$$\mathrm{nDCG_p} = \frac{DCG_p}{IDCG_p},$$



1. Järvelin, Kalervo, and Jaana Kekäläinen. "Cumulated gain-based evaluation of IR techniques." ACM Transactions on Information Systems (TOIS) 20.4 (2002): 422-446.

Learning to rank approaches

Learning to rank - pointwise[1] approaches

- We train a classifier and treat outputs as document scores.
- No context of document listing – as such, no direct comparison of documents.
- Not optimising NDCG, even though it has been shown[2] that perfect classification leads to perfect ranking (perfect NDCG).

1. Burges, C. J. (2010). From ranknet to lambdarank to lambdamart: An overview. Learning, 11(23-581), 81.
2. Li, P., Wu, Q., & Burges, C. J. (2008). Mcrank: Learning to rank using multiple classification and gradient boosting. In Advances in neural information processing systems (pp. 897-904).

Learning to rank - pairwise approaches

- Example of RankNet[1] – neural net to express scoring function.
- With pairwise optimisation[2] we look at probabilities of choosing one doc *Ui* over another *Uj*:

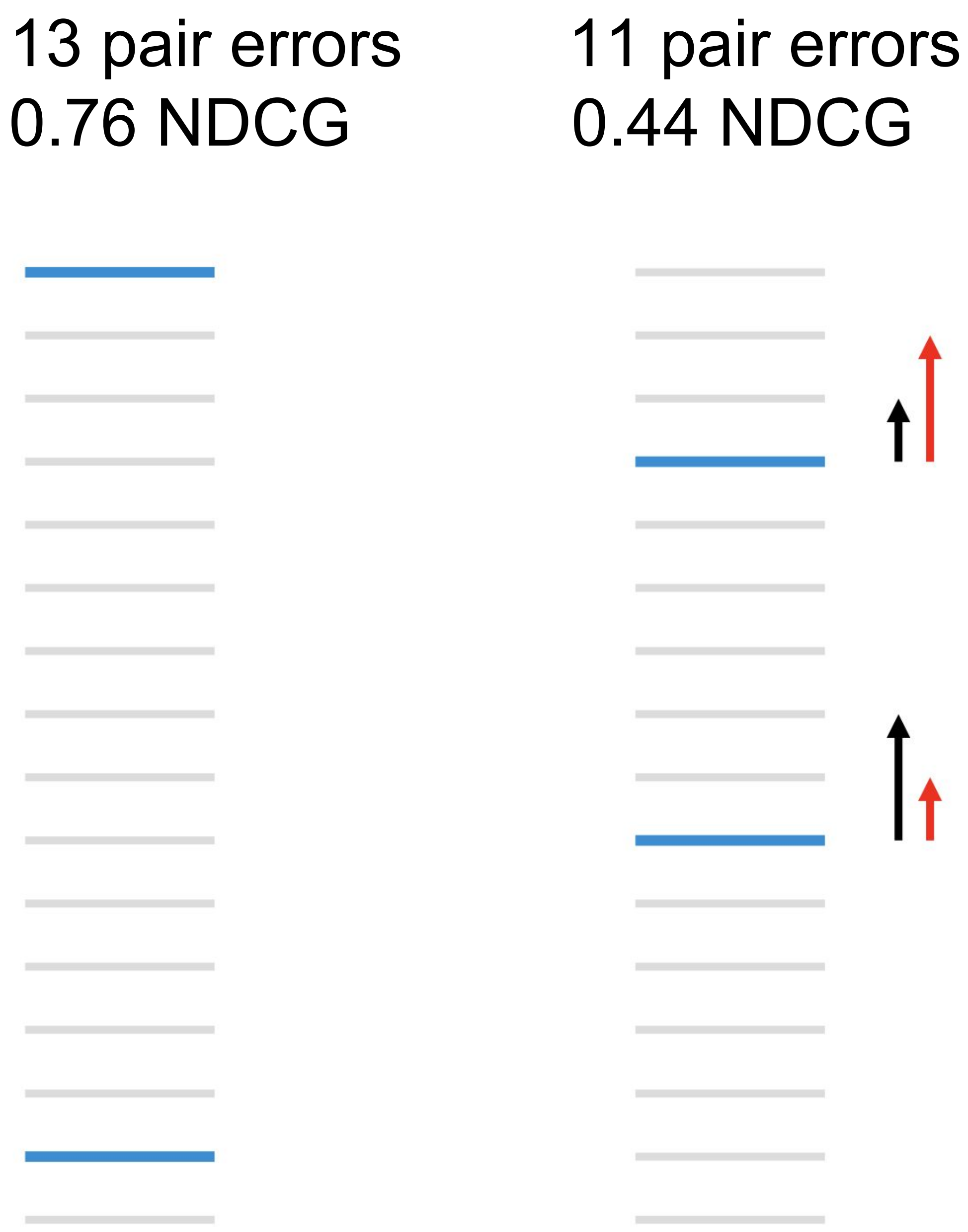
$$P_{ij} \equiv P(U_i \triangleright U_j) \equiv \frac{1}{1+e^{-\sigma(s_i - s_j)}}$$

- We optimize cross-entropy:

$$C = -\bar{P}_{ij} \log P_{ij} - (1 - \bar{P}_{ij}) \log(1 - P_{ij})$$

1. Burges, C. J. (2010). From ranknet to lambdarank to lambdamart: An overview. Learning, 11(23-581), 81.
2. C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier ,M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In Proceedings of the 22nd International Conference on Machine Learning, 2005.

Learning to rank - listwise approaches



Learning to rank - listwise approaches

- In the perfect world we would like to optimize NDCG directly but that’s impossible.
- LambdaRank[1] overcomes this problem in a way that cost of every pair is weighted by the change of NDCG given the pair is swapped

$$\lambda_{ij} = \frac{\partial C(s_i - s_j)}{\partial s_i} = \frac{-\sigma}{1 + e^{\sigma(s_i - s_j)}} |\Delta_{NDCG}|$$

1. Burges, C. J. (2010). From ranknet to lambdarank to lambdamart: An overview. Learning, 11(23-581), 81.

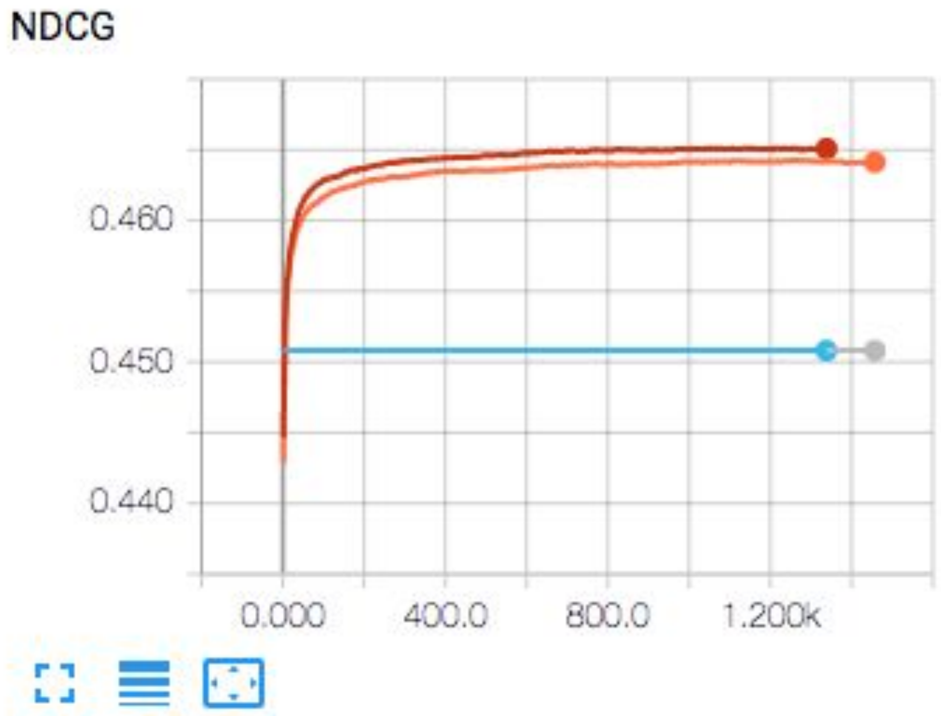
Learning to rank implementation @ allegro.pl

Collecting training data

- Dataset containing: *context*, *offer features*, *list of actions* (clicks, transactions etc.)
- We needed to join sources like:
  - Offer features and custom offer aggregates
  - User features
  - Search events
  - Frontend events
- Aggregated data sources speed up experimentation.
- Currently we gather approximately 200M observations daily.

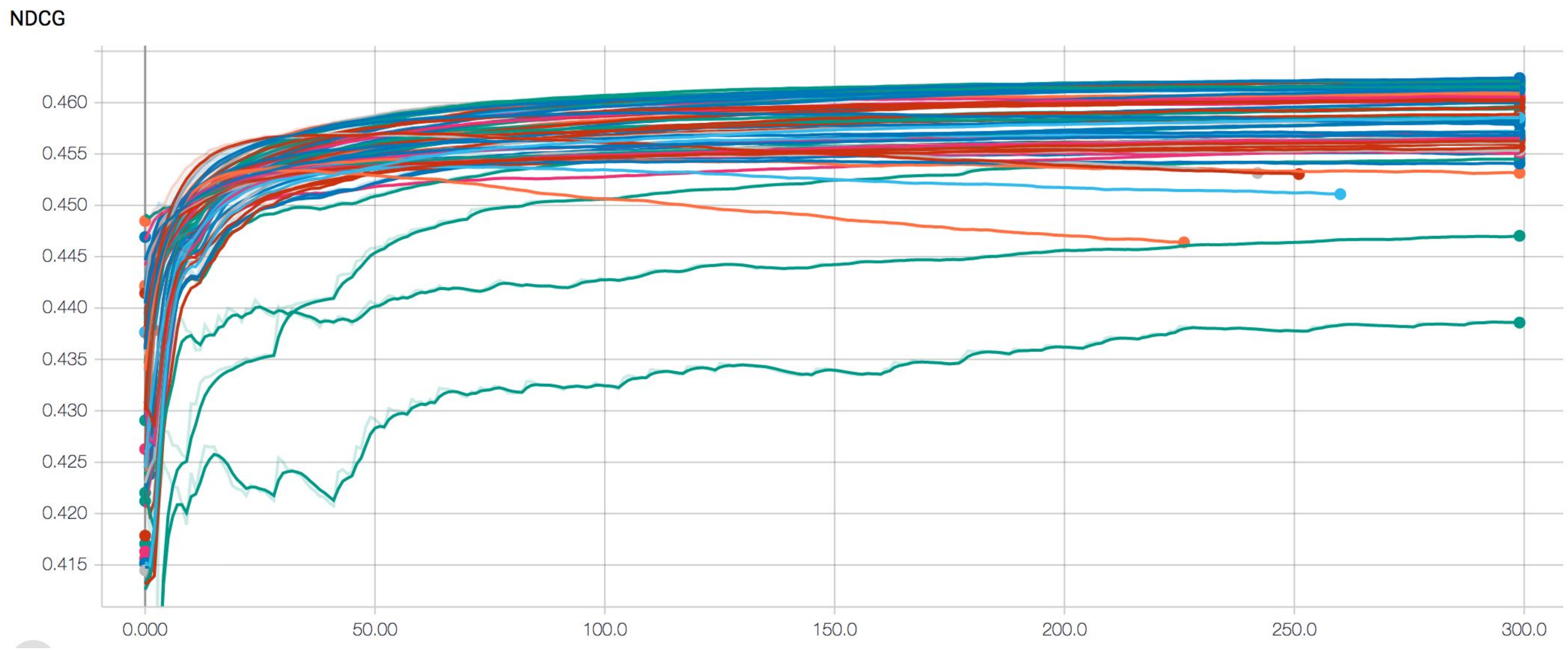
Training

- Currently the underlying model is an XGBoost trees model trained with loss *rank:pairwise*.
- We made a suite of Python packages for dataset sampling, model training and validation.
- This allows for easy parallelization.
- During the training procedure we keep our results in tensorboard.



Hyperparameter tuning

- XGBoost has several important hyperparameters.
- Google vizier[1] - GCP hosted hyperparameters tuning service.
- Take aways:
  - Early stopping speeds up the process.
  - Small models can be as good as large given good hyperparams.
  - Be aware of XGBoost quirks.



1. Golovin, Daniel, et al. "Google vizier: A service for black-box optimization." Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2017.

Feature importance

- XGBoost (and any complex model) is hard to interpret.
- Methods like: *cover*, *weight*, *gain*, did not correlate well with NDCG gains.
- We investigated two more advanced methods:
  - LIME[1] (Local Interpretable Model-agnostic Explanations).
  - SHAP[2] (SHapley Additive exPlanations).

1.Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "Why should i trust you?: Explaining the predictions of any classifier." *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2016.

2. Lundberg, Scott M., and Su-In Lee. "A unified approach to interpreting model predictions." *Advances in Neural Information Processing Systems*. 2017.

Presentation Biases

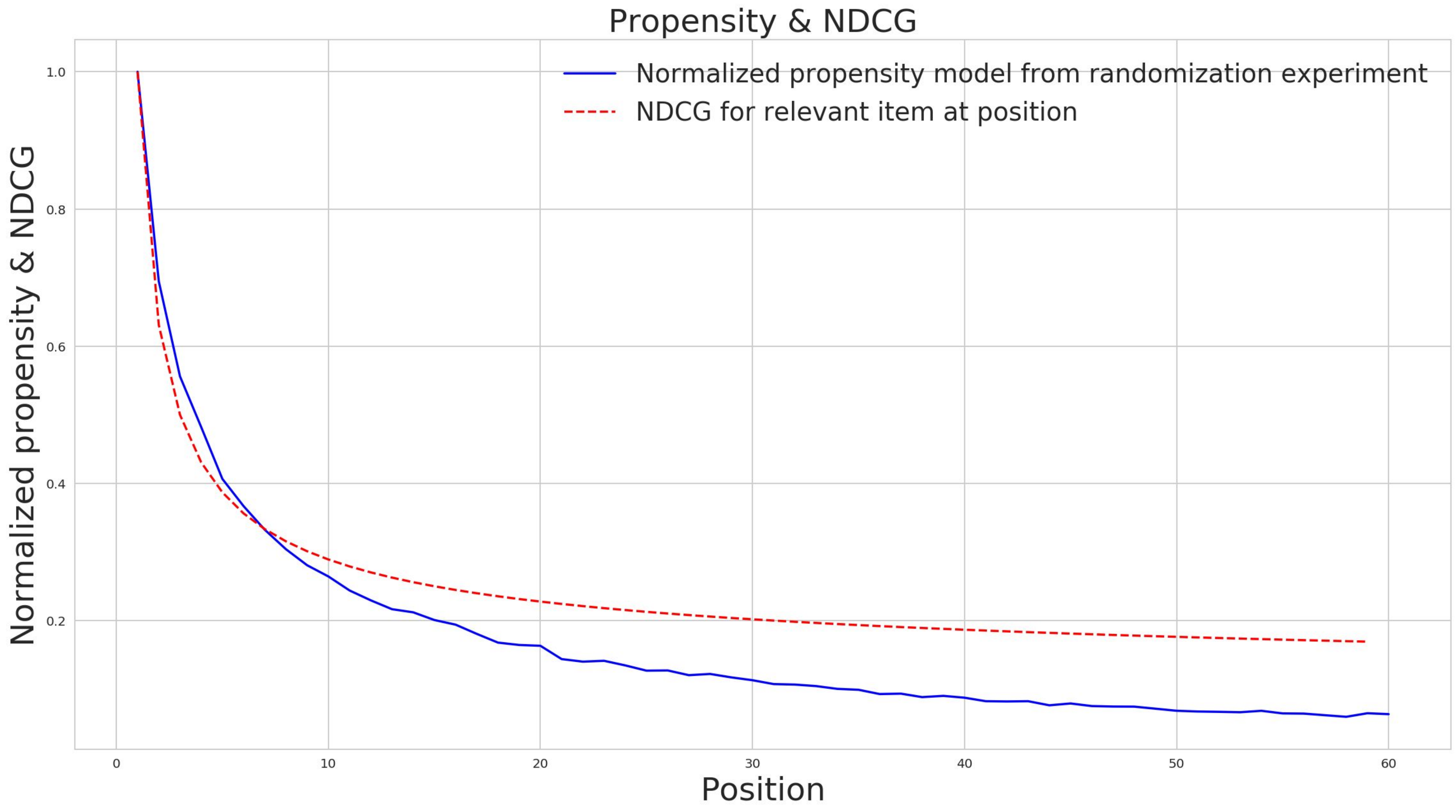
- Our platform is not free from position bias and other presentation biases.
- To fight bias we need to understand it and de-bias the data for training.
- Basic approach presented in[1]:
  - Calculate Position-based propensity model which assumes a following observation model:

$$P(\mathrm{e}_i(y) = 1 | \operatorname{rank}(y|\bar{\boldsymbol{y}})) \cdot P(\mathrm{c}_i(y) = 1 | \mathrm{r}_i(y), \mathrm{e}_i(y) = 1).$$

  - Calculated by randomization or swapping intervention.
  - Inversed Propensity Scoring is used to weight training examples.

1. Joachims, Thorsten, Adith Swaminathan, and Tobias Schnabel. "Unbiased learning-to-rank with biased feedback." *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 2017.

Position Propensity



High dimensional data in GBT models

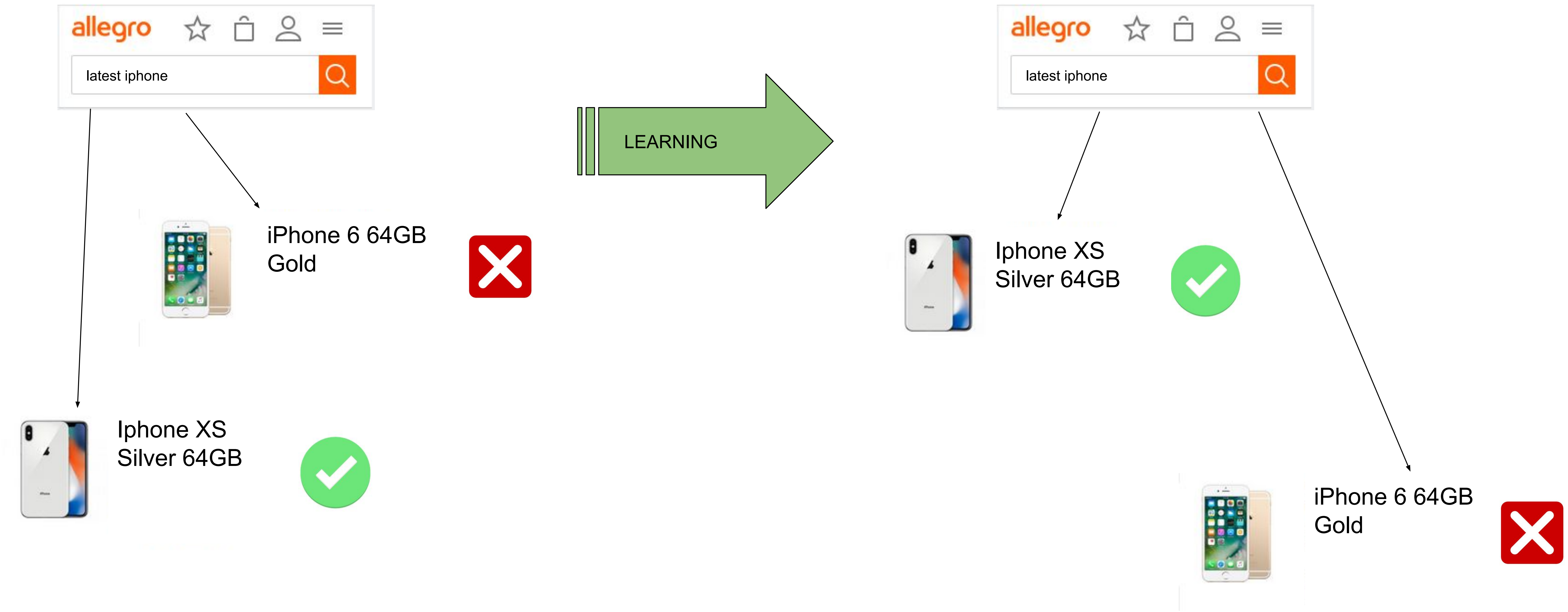
How about incorporating high dimensional data in GBT models?

- GBT model limits our abilities of rich data exploitation – like text or images.
- End-to-end, SGD-based approach is one of our future experiments.
- We create similarity features from above-mentioned modalities with deep learning.
- We match search phrases with image data or textual data by representation learning.
- Modeling objective: maximise similarity of matching entities.
- Cosine/euclidean distances between vector representations work as features.

Incorporating textual data into GBT models

- We've tested several approaches to learning text representations.
  - Distributed representations trained with methods following distributional hypothesis (like Word2Vec[1] or GloVe[2]).
  - Starspace[3], library developed by FAIR.
- We minimise cosine distance between positive title/phrase pairs, maximising between negative ones.

1. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).
2. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).
3. Wu, L., Fisch, A., Chopra, S., Adams, K., Bordes, A., & Weston, J. (2017). Starspace: Embed all the things!. arXiv preprint arXiv:1709.03856.

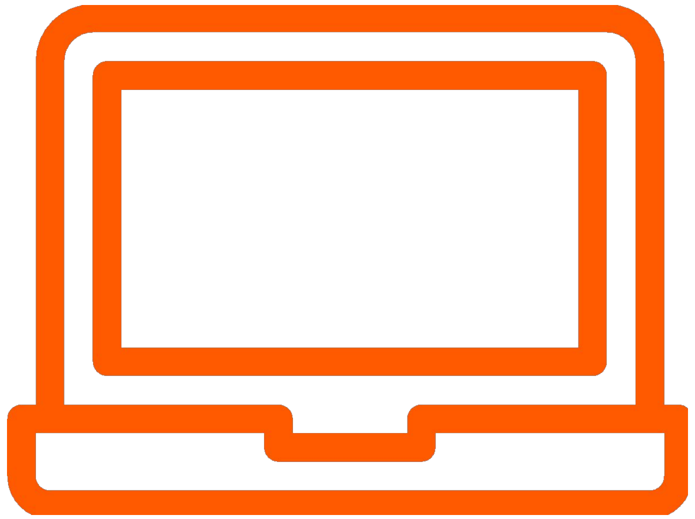

allegro
latest iphone
iPhone 6 64GB Gold
Iphone XS Silver 64GB
LEARNING
allegro
latest iphone
Iphone XS Silver 64GB
iPhone 6 64GB Gold

Incorporating image data into GBT models

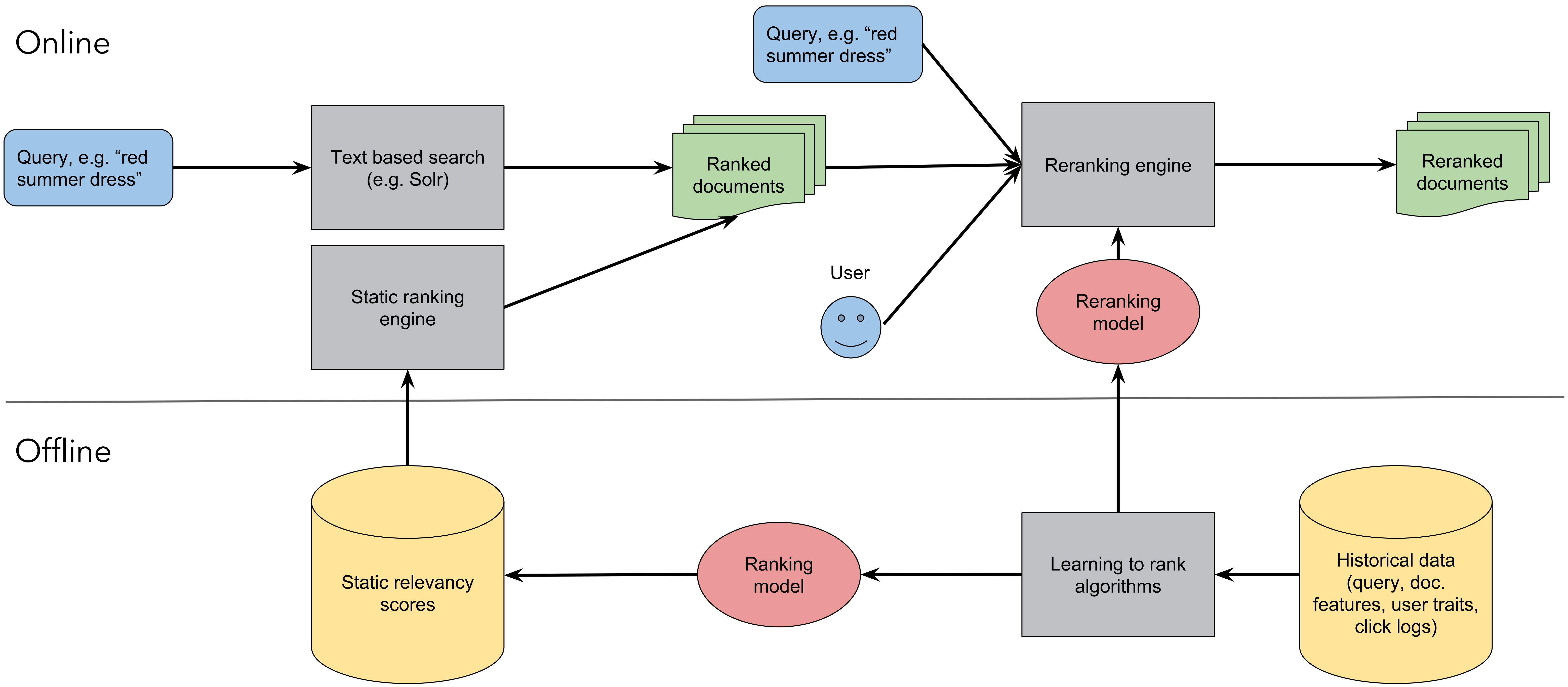
- Baseline: high level activations from pretrained ImageNet models (InceptionV3 and VGG16).
- Let’s train general and semantic image representation.
- We can use contextual data in a supervised setting.
- Unsupervised learning methods (namely Adversarially Learned Inference[1] model and beta Variational Autoencoders[2]) have proven to be superior to baseline.

1. Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M., & Courville, A. (2016). Adversarially learned inference. arXiv preprint arXiv:1606.00704.
2. Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., ... & Lerchner, A. (2016). beta-vae: Learning basic visual concepts with a constrained variational framework.

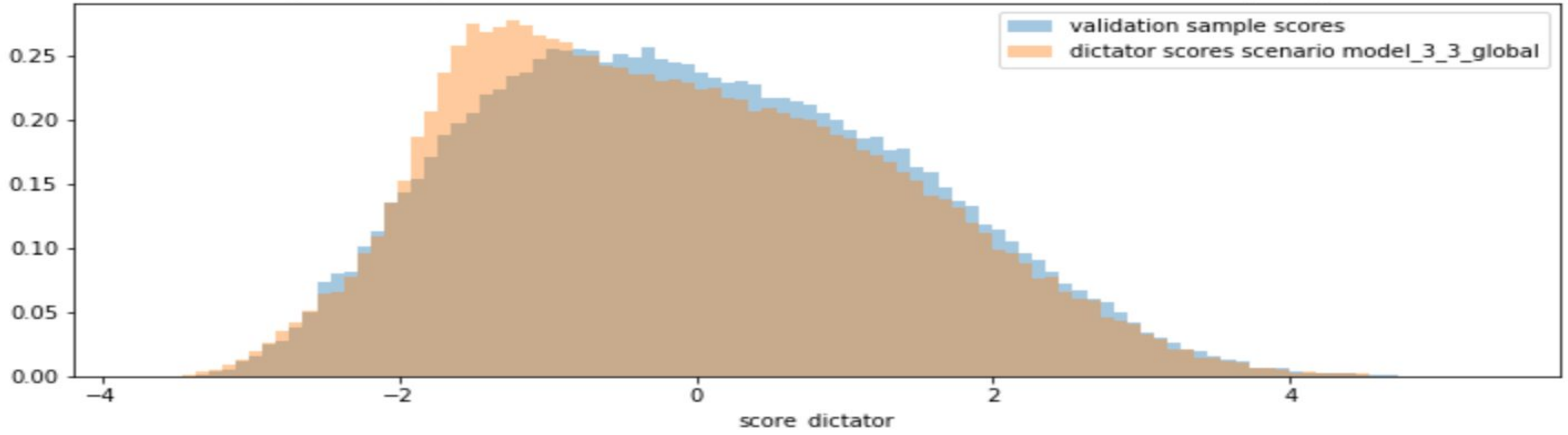
Deployment



Ranking in large scale search engines

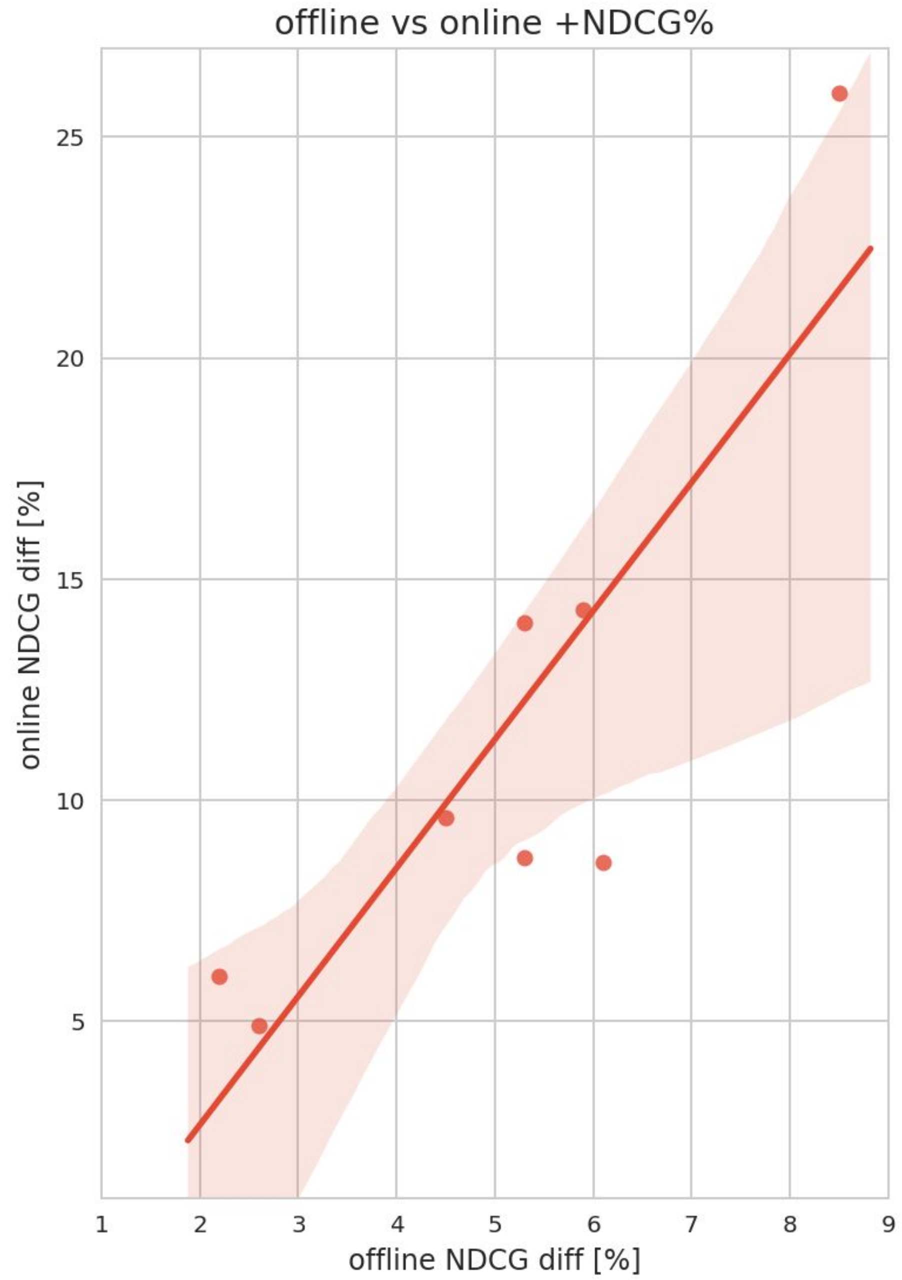


Deployment verification



What happens when we run our models online?

- When training we observed up to 9% NDCG improvement.
- Same model deployed to production gave bigger improvement.
- Finding a correlation between offline and online metric is an important step in such project.
- Bigger improvement in online can be justified by:
    - training on the results of previous ranker.
    - position bias.



Thanks!

Works presented in this talk have been developed by several teams at Allegro.pl.