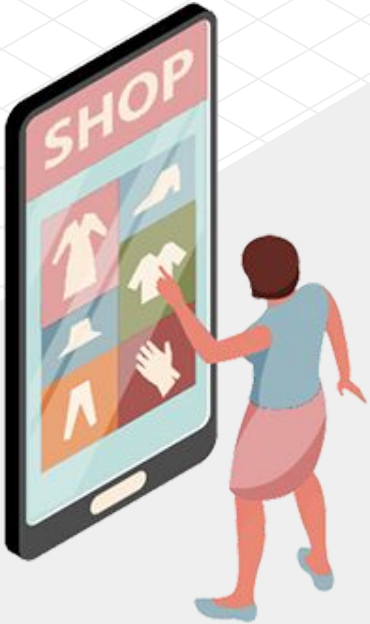
Click-Through Rate Prediction with Neural Recommender Systems

Kevin Antony, Utkarsh Lal, Bonnie Liu,
Nisha McNealis, Siddharth Nandy

CS 247 Winter 2024 (Group 3)

01

Introduction



Background

- CTR prediction plays a crucial role in digital advertising and recommender systems, by determining the likelihood of a user clicking on an ad or recommendation.

- Early models often fall short in capturing complex nonlinear relationships and interactions between users and items.

- By focusing on neural recommender systems, we aim to contribute to the ongoing evolution of CTR prediction methodologies, offering insights and practical solutions to the challenges faced by the industry.

Terminology

- **Impression**: every time an ad/item is shown to a user
- **Syndicator**: advertiser/sponsor of an item
- **Campaign**: set of items promoted together by a syndicator

CTR Prediction

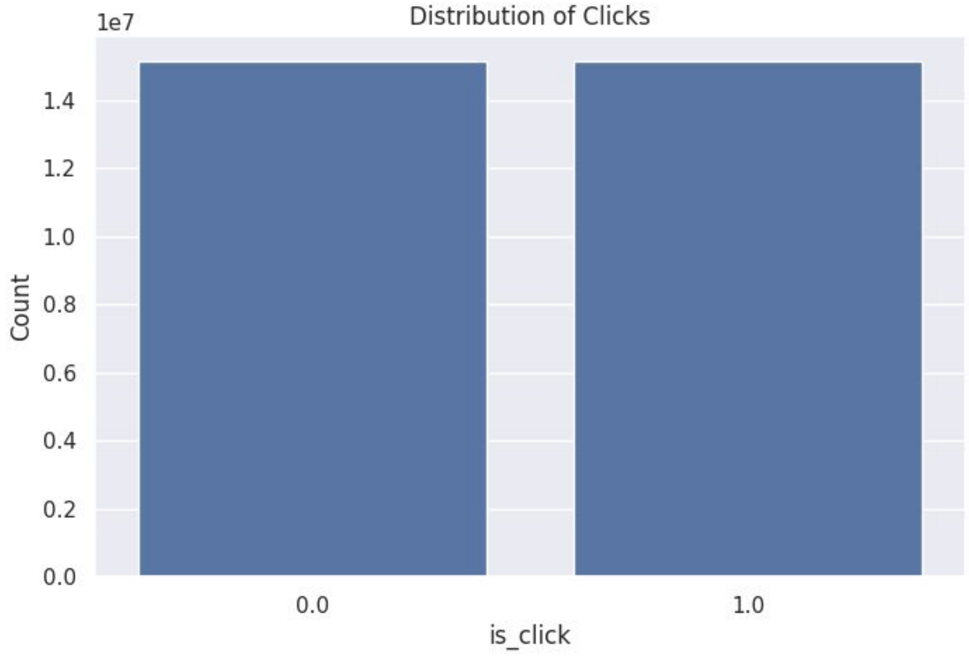
- **Click-through rate**: Number of clicks that an item receives divided by the number of times it is shown ("**impressions**")
    - Critical metric in digital advertising and content distribution
    - Sponsored items are provided by **syndicators** (advertisers)
- Challenges:
    - Sparse and imbalanced data: each user only clicks on a small number of ads/items
    - New items/users are difficult to make predictions for
    - User behavior can change over time

CTR Prediction

- Techniques for CTR prediction range from very simple to extremely complex
  - Logistic Regression
  - Decision Trees
  - Random Forests
  - Deep Interest Networks (DINs) *(Zhou et al. (2019))*
  - Feature Generation by Convolutional Neural Networks (CNNs) *(Liu et al. (2019))*
  - Modeling Feature Interactions via Graph Neural Networks *(Li et al. (2019))*

Dataset

- Labeled dataset provided by Taboola
- 30,253,436 useable samples
- 16,263,893 users
- 95,073 items
- 23 features
- **Balanced**



Features: User

- **user_id_hash:** unique identifier of a specific user
- **user_target_recs:** the number of times the user saw this target
- **user_recs:** total number of items presented to the user so far
- **user_clicks:** total number of items clicked by the user so far

Features: Item

- **target_id_hash**: unique identifier of a specific item
- **syndicator_id_hash**: unique identifier of the syndicator of the item
- **campaign_id_hash**: unique identifier of the item’s campaign
- **empiric_calibrated_recs:** the number of impressions of the current item, normalized
- **empiric_clicks** - the number of clicks on the target item
- **target_item_taxonomy**: third party hierarchical taxonomy (category) of the target, eg. Business, Entertainment, etc.
- **placement_id_hash**: location in the publisher site in which the ad has been viewed
- **page_view_start_time**

Features: Context

- **publisher_id_hash:** unique identifier of the source's publisher
- **source_id_hash:** unique identifier of the current source
- **source_item_type:** the type of the source item, eg, Text, Video etc.
- **browser_platform:** desktop, mobile, tablet, etc.
- **os_family:** Windows, iOS, etc.
- **country_code:** eg. US
- **region:** eg. FL, TX
- **day_of_week**
- **time_of_day**
- **gmt_offset**

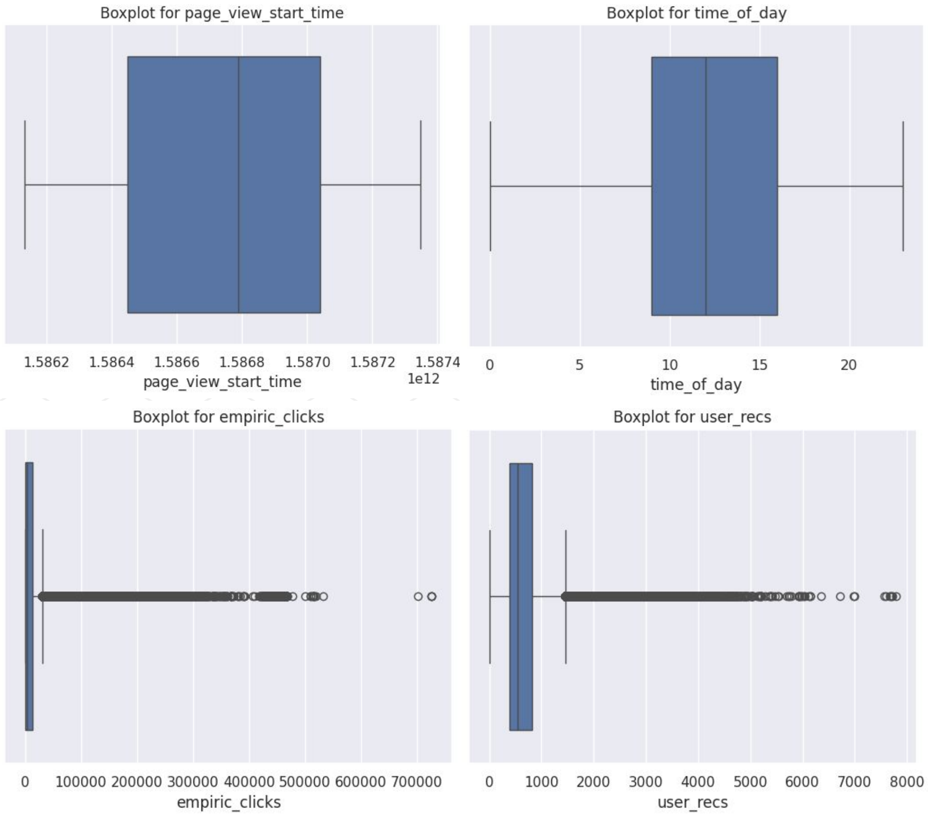
Features

- **More important features (based on Random Forest):**
  - empiric_calibrated_recs
  - empiric_clicks
  - page_view_start_time
  - user_recs
  - user_clicks
- **Less important features (based on Random Forest):**
  - os_family
  - day_of_week

02

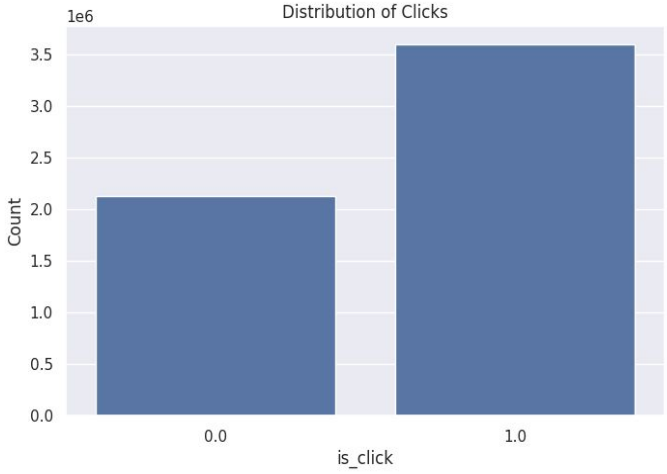
Methodology

Dataset Outliers



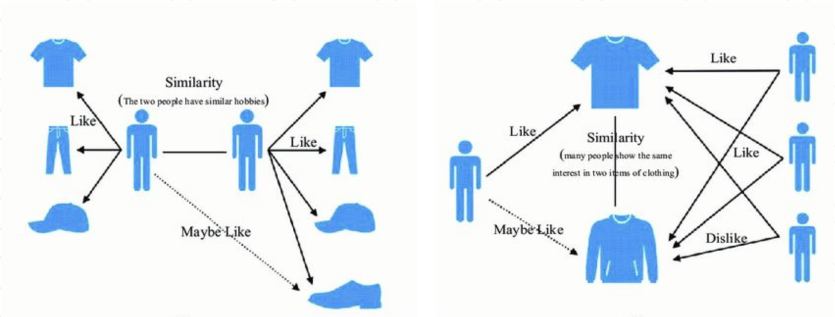
Using a Subset of the Data

- Dataset is sparse; 16 million users with at least 5 million who only have interactions with 1 item
- Subset used for model: users who interact with **at least eight items**
    - Undersampled to account for imbalance in clicks and non-clicks
    - Number of users: **462,030**
    - Number of items: **55,876**
    - Number of user-item pairs: **4,246,524**



Initial Directions

- Wanted to model how user-item relationships impacted CTR
- Initially used Collaborative Filtering
  - The issue is CF doesn't take into account external sources of information
- To account for the additional information found in our feature set we also explored various Hybrid Models that combined user-item interactions with other information found in our data which we believed would yield the best results



(a) (b)

Model Architectures

- **SVD**: Matrix factorization to uncover latent factors related to user and item interactions.
- **SVD++**: Extension of SVD that incorporates implicit feedback.
- **WideDeep**: Combines “wide” memory of interactions & “deep” neural network for generalization of feature combinations.
- **DeepFM**: Captures interactions for CTR prediction through recommendation with deep learning for feature learning.
- **Neural Collaborative Filtering (NCF)**: combining collaborative filtering methods with neural network.
- **Transformer**: Self-attention that enable the model to weigh the input data, used for various sequence-to-sequence tasks.

03

Results & Findings

Baselines

Baseline Method	Accuracy
Logistic Regression	0.50
Decision Tree	0.66
Random Forest	0.73
Perceptron	0.68

LibRecommender

- LibRecommender is an easy-to-use recommender system focused on end-to-end recommendation process. It lets users quickly train and deploy different kinds of recommendation models.
- It was built to predict ratings or rankings, so here, we modified it to treat is_click as a rating: either 0 or 1.

Model Architecture	**Accuracy**
SVD	0.719
SVD++	0.726
WideDeep	0.731
Neural CF	0.734
DeepFM	0.735
Transformer	0.741

https://pypi.org/project/LibRecommender/

04

Challenges

Dataset

- Huge dataset required a lot of computational power
- We found that most of the users in our dataset have only one item mapped to them once ever making ideas like *collaborative filterin*g a weak idea for creating recommendations.
- The target data (is_click) was binary; thus, we had to figure out a way to modify the rating and ranking systems to work accordingly

Models

- We attempted to create our own NCF model tailored to the problem
    - Idea: learn user & item embeddings using CF, incorporate user, item, and context features in neural network
    - Could not figure out reason for poor performance in time
- Looking for a good open source model was challenging since we had to fit our data to match the format of each open-source model
- We tried many recommendation libraries like Spotlight, Tensorflow recommenders and DeepCTR which weren’t up to mark in performance

05

Conclusions

- Demonstrated the application of neural collaborative filtering to predict click given a user and corresponding item on a dataset with over 400K users, achieving near 75% accuracy.
- Architectures like DeepFM and Transformers gave the best result in predicting the likeliness if an advertisement will be clicked by a user
- Gained insights into the challenges and practicalities of deploying scalable recommender systems
- Addressed computational constraints and data sparsity issues through effective data preprocessing and model tuning.

Final Thoughts: Our journey highlights the dynamic nature of data, showcasing both the power of neural approaches and the constant need for innovation in the field of ads

Future Work

- Investigate graph-based neural network models to better handle user-item interaction complexity and sparsity
- Explore context embeddings to personalize recommendations
- Continuous model refinement with emerging DL techniques
- Ensemble techniques to combine models to leverage the strength of each one of them
- Expand computational resources for increased experimentation.

FAQs

Q. What is Click-Through Rate Prediction?
A. CTR Prediction involves estimating the likelihood that an ad or recommendation will be clicked on by users

Q. Is the project predicting for new or existing users
A. Existing Users

Q. What is the significance of the dataset we used from Kaggle?
A. The dataset provided a realistic scenario for CTR prediction, containing a huge rich set of user-item interaction data.

Q. How did we use class learnings in this project?
A. Our brainstorming was based around collaborative filtering which we used to branch out to think of new ideas and architectures giving a thorough understanding of its principles and practicality.

References

- Zhou, G., Mou, N., Fan, Y., Pi, Q., Bian, W., Zhou, C., Zhu, X., \& Gai, K. (2019). Deep interest evolution network for click-through rate prediction. In Proceedings of the AAAI conference on artificial intelligence (AAAI-19). 33(01), 5941–5948. https://doi.org/10.1609/aaai.v33i01.33015941.
- Bin Liu, Ruiming Tang, Yingzhi Chen, Jinkai Yu, Huifeng Guo, and Yuzhou Zhang. 2019. Feature Generation by Convolutional Neural Network for Click-Through Rate Prediction. In The World Wide Web Conference (WWW '19). Association for Computing Machinery, New York, NY, USA, 1119–1129. https://doi.org/10.1145/3308558.3313497
- Zekun Li et al. Fi-gnn: Modeling feature interactions via graph neural networks for ctr prediction. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management (Nov. 2019). 10.1145/3357384.3357951. http://dx.doi.org/10.1145/3357384.3357951.
- Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In Proceedings of the 26th International Conference on World Wide Web (WWW '17). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 173–182. https://doi.org/10.1145/3038912.3052569

Thank You!