Enhancing User Experience by Integrating Real-Time Search Data with Historical Data to Personalize Recommendations for a Timeshare Exchange Company

Hemanya Tyagi
Krannert School of Management
Purdue University
West Lafayette, USA
tyagih@purdue.edu

Robin Jindal
Krannert School of Management
Purdue University
West Lafayette, USA
rjindal@purdue.edu

Dibyamshu Shrestha
Krannert School of Management
Purdue University
West Lafayette, USA
dshresth@purdue.edu

Mohinder Pal Goyal
Krannert School of Management
Purdue University
West Lafayette, USA
goyal62@purdue.edu

Matthew A. Lanham
Krannert School of Management
Purdue University
West Lafayette, USA
lanhamm@purdue.edu

***Abstract*— With companies focusing intensively on customer experience, personalization and platform usability have become crucial for a company's success. Hence, providing appropriate recommendations to users is a challenging problem in various industries. According to Salesforce, personalized product recommendations drive just 7% of visits but 26% of revenues on an e-commerce website [1]. Recommendations play an important role in other industries as well. For instance, two out of three movies watched on Netflix are recommended [2].**

We work towards enhancing the recommendation system of a timeshare exchange platform by leveraging real-time search data. Previously, the recommendation model utilized historical transactions, searches, resort data, and member data to recommend resorts to users. This model was deployed online through a batch process once a day. The limitation of this model was that it did not consider the real-time searches of the user, hence losing context. This directly impacted the click-through rate of the recommendations, and the users had to navigate the website excessively to find a satisfactory resort. We build a model such that it utilizes not only the historical transactional and master data but also the real-time search data to provide multiple relevant resort recommendations within five seconds.

To develop the recommendation system, we develop models that diversify the recommendations and overcome the problems of redundancy and irrelevancy. This is done by architecting the models in a way that penalizes older searches and values recent searches using a decay function. The models use matrix factorization, collaborative filtering and cosine similarity to personalize recommendations and account for factors such as resort amenities, search history and popularity of resorts. (*Abstract*)

Keywords—recommender system, collaborative filtering, matrix factorization, cosine similarity, real-time search

I. Introduction

Businesses integrate recommendation systems on their websites to ease the search for the users. In fact, a research by Engagement Labs claims that personal recommendations are the #1 driver of consumer purchase decisions at every stage of the purchase cycle, across multiple product categories [3]. Furthermore, one can argue that the benefits of a recommendation system outweighs the cost for creating and maintaining recommendation systems [4]. However, deploying a recommendation system in the production environment comes with a variety of challenges. For instance, a recommendation system needs to be dynamic for being accurate and efficient. Moreover, by being dynamic it should incorporate the context of the user search at every instance. Other common issues are optimizing response times, frequently updating models, and predicting based on unseen data, also commonly known as the cold-start problem [5].

In this paper, we present the idea to incorporate real-time search history in considering recommendations. This would allow the website to learn the context of what user wants and recommend resorts for them based on their past booking preferences and filtered by the current real-time search. This would allow website to recommend appropriate products/services for the client which would increase client satisfaction.

We specifically work on recommending resorts for a timeshare exchange company. Currently, a recommendation model is used by the company to recommend relevant resorts to the user based on the member data, resort amenities and other factors. The model trained thus is deployed to the website, once a day. Hence, if a user searches for a resort in Denver today, the model can recommend resorts in Chicago, which is based on the search history of the user up till yesterday. If the user further filters the search to Aurora, Denver, the recommendation system would still recommend the resorts recommended in the previous search. This shows that the recommendation system is greatly affected by this inaccuracy. The context of the current search is a significant factor in determining which resort the user will go to next. Also, since the recommendations happen in real-time, the recommendation model should return results in a matter of seconds. Therefore, speed also becomes a crucial factor in recommending relevant resorts. We work towards designing a model which incorporates the real-time search history and recommends eight relevant resorts to the user within five seconds.

Traditionally recommendation systems have used methods like collaborative filtering and RankBoost algorithms as their go-to methods. However, collaborative filtering suffers from a few problems, the most prominent of which is the cold-start problem. To overcome this, hybrid models have been used in the past. Still, there is little research on how to use real-time search data of users to recommend a relevant resort in a time-share industry. The problems are not just limited to which algorithms to choose, but how to measure the performance of the offline model and how to record and accommodate the implicit feedback of the user.

To develop the recommendation system, we develop models that diversify the recommendations and overcomes the problems of redundancy and irrelevancy. This is done by architecting models in a way that selectively values the recent searches and penalizes the older searches.

The rest of the paper is divided into the following sections: The Data section is akin to a data dictionary and describes each variable in detail. Methodology describes the experimental design. Model section elaborates upon our model designs and evaluation parameters. Then the Results section compares the performance of ours with the baseline model. Conclusion throws the light on the applications of our research and further scope. Finally, References section contains the research we referred.

II. DATA

In this study, we used the search and confirmation data provided by the client. The search data consists of searches conducted by around 1.3 million members in the past year, i.e., from January 2019 – January 2020. The dataset has a record of about 159 million searches in the past year. The search data has information on search details like date and time of search as well as more information about the region that was searched like region, destination, city, or resort ID of the search. It also has the information on the month and specific date the member is searching for.

The other primary table for this project is the confirmation data table, which has details about the bookings by around 1 million members for around 6000 resorts in the past five years. There are approximately 5 million recorded confirmations from January 2015 to January 2020. The dataset has information about the resort ID and resort amenities that a member has booked. It has details about when the booking was confirmed and the actual start and end date of the booking. The dataset also contains a cancellation index, which indicates if a person made a booking and later canceled it.

The principal table for our model on cosine similarity is resort amenities in addition to the tables mentioned above. The resort amenities table has information about 350+ features of all the resorts. Since the online portal has information in 17 different languages, we used tables that help us to convert language centric IDs to language-independent IDs so that the same resort in different languages has the identical IDs after using these tables. Availability information of resorts are also provided in the dataset but hasn't been used.

Table 1 provides a brief description of the different tables used in this study.

TABLE 1. DATA DESCRIPTION

Table No.	Table Name	Description
1	Confirmation Data	Booking data for 1M members in the past 5 years
2	Search Data	Search data for 1.3M members in the last year
3	Resort amenities	350+ amenities information of 20,000 resorts
4	Check in Month Table	Covert check in month ID in different languages to Year-month of check in
5	Member Data	Information of 2.46 Million members
6	Region ID Table	Convert Region ID in different languages to Region ID independent of language and get details about the region
7	Destination ID Table	Convert Destination ID in different languages to Destination ID independent of language and get details about the destination
8	City ID Table	Convert City ID in different languages to City ID independent of language and get details about the City
9	Resort to XML	Information about all the resorts mostly its locations and all the associated IDs for that resort
10	Available data	Availability and details of 4061 resorts

III. METHODOLOGY

The prime objective of the experiments is to find an algorithm which captures user preferences and recommends relevant resorts based on that. To develop the models, we need to investigate the following questions: What factors are users looking for while booking a resort? How to recommend resorts to first-time users who have no confirmation or search history?

The methodology used to answer the above questions is extensively described in Fig. 1.

A. *Explanatory Data Analysis (EDA)*

The first step towards solving any data problem is visualizing the data and finding insights which might be useful in the modeling process. Some interesting insights that we got from the EDA are:

- The distribution of the confirmation data is right skewed in which more than 25% of the users have booked resorts less than three times till date.

- Two resorts in Kissimmee and Las Vegas are the most booked resorts with more than 26,000 bookings combined.

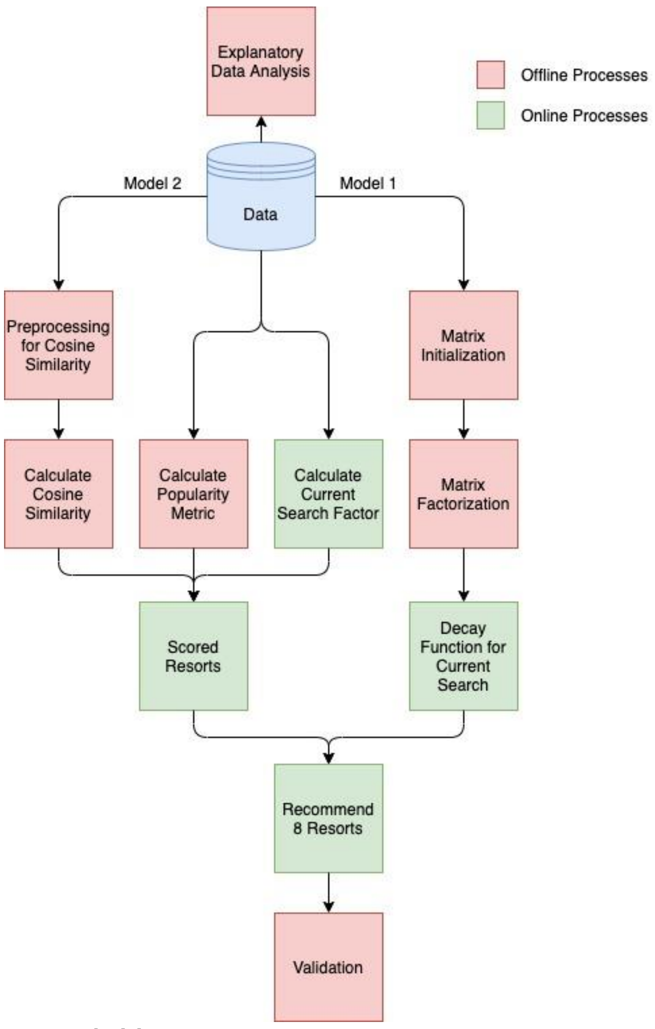
Explanatory Data Analysis
Offline Processes
Online Processes
Model 2
Data
Model 1
Preprocessing for Cosine Similarity
Matrix Initialization
Calculate Cosine Similarity
Calculate Popularity Metric
Calculate Current Search Factor
Matrix Factorization
Scored Resorts
Decay Function for Current Search
Recommend 8 Resorts
Validation

Fig 1. Methodology

- There is seasonality in the data with most bookings being done in quarter 3. Furthermore, most users book resorts for 7 days.

B. *Data Preprocessing*

Preprocessing was done only for Model 2. Since Model 2 is based upon cosine similarity of resort amenities, resort amenities table had to be preprocessed.

- Column Selection: There are 165 columns in resort amenities table, out of which several columns have majority of data missing. We selected 17 columns from this table in which majority of the data was not missing.

- Missing Data Handling: For categorical columns, a new dummy encoded column was created to handle missing values. For the numerical column, Golf Distance, we substitute the missing values with 0, if golf is onsite or nearby, 200 otherwise.

- Min-Max Normalization: Since Golf Distance can cause a significant bias while calculating cosine if not normalized, Golf Distance was normalized using a min-max normalizer.

C. *Modeling*

We experimented upon two modeling approaches. The first model (**Model 1**) assumes that there are similarities amongst users and similarities amongst resorts and recommends resorts to user based on these similarities. The second model (**Model 2**) assumes that users book resorts on the basis of resort popularity, amenities and search behavior.

D. *Validation*

- Data Partitioning: We run two models: model 1 which is based on cosine similarity, and model 2 which is based on matrix factorization. Since matrix factorization is a computationally intensive process, the train-test split for both the models must be different.

- For the first model, we selected 3 sets of 1600 members who had 2 or more bookings. Then for each member, we split the data till the point of their second last booking and ran Matrix Factorization algorithm. For test, we selected member having 2 or more bookings in 2019 and recommended resorts to them based on each search.

- For the second model, we take a sample of 100 users who have more than 2 confirmed bookings between October 1, 2019 and December 31, 2019. We take a test window from October 1, 2019 to December 31, 2019. For each day, t in the test window, we run the model with confirmation data up to t-1 days and all search data available till day t. We then predict the resorts for each search on day t.

- Success Criteria: To evaluate and compare the two models, we first need to define success. If a user books a resort that was recommended to him/her within 17 days prior to the confirmation date, we define it as a success.

- Evaluation Metric (Accuracy): The evaluation metric then can be defined simply as the ratio of the number of successes and the number of bookings done by the user.

IV. MODELS

A. *Model 1*

Model 1 incorporates the technique of Matrix Factorization to recommend resorts.

Matrix Factorization: Matrix Factorization is the technique to predict the missing values in a sparse matrix. As the name suggest, it decomposes a matrix into product of two matrices.

In Fig 2., Matrix X is the product of Matrix A and B. Matrix decomposition is nothing but the process of factorizing X into A and B.

Fig. 3. shows the need of matrix factorization. We use matrix factorization when we need to fill in missing values in a matrix. This is specifically needed in recommendation systems, when we do not have the data about how much a user will value a resort, if s/he has not yet searched for or booked that resort.

Initially each search or booking adds points to a user-resort cell in the matrix. For example, if a member searches for a resort in Colorado, Denver then for that member, all the resorts in that area will be given some points, that is, Search Factor. The formula used to allocate points for resorts based on member's searches and bookings were created for this purpose and tested.

$$\textit{Search Factor} = (\textit{Surge Factor}) * \left(\frac{a^x - a^{-x}}{a^x + a^{-x}}\right) * \left(\frac{x}{y}\right)$$

Here,

y = Average number of searches per booking of the user per search type

x = Number of times the transaction type is done

Surge factor can be computed from Table 2.

"a" can be computed from Table 3.

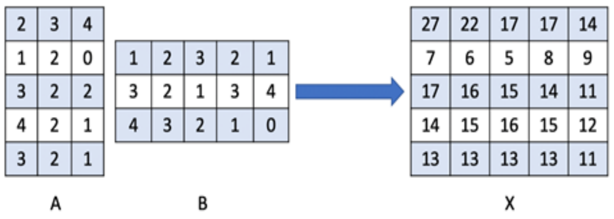

Fig 2. Matrix multiplication

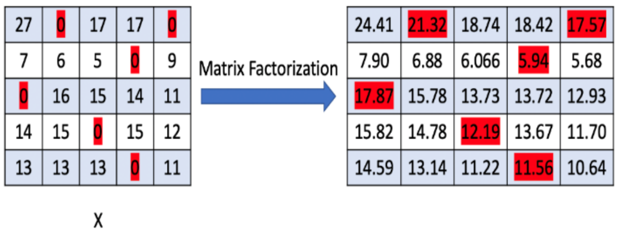

Fig 3. Matrix factorization

TABLE 2. SURGE TABLE

Transaction Type	**Definition**	**Surge Factor**
Bookings	All historical bookings	5
Today's Search	Searches done today	4
Recent Search	Searches done after last booking	2
Past Search	All searches done before last booking	1

TABLE 3. WEIGHTS FOR SEARCH TYPE IN MODEL 1

Search Type	**a**
Region Search	1.05
Destination Search	1.1
City Search	1.5
Resort Search/Booking	2

Matrix Initialization and Factorization: Resorts that do not show up in member's searches or bookings are populated as 0 in the matrix initialization process. Using all transaction types except Today's Search, the matrix is populated using matrix factorization with all available resorts as column and members as rows. The matrix factorization model is run once per day.

Recommending Using Matrix Factorization. Using the output of Matrix Factorization, when a member searches a region in the present day, Search Factor will be added to the resorts in that region and recommendation will be given as per the criteria in Fig. 4.

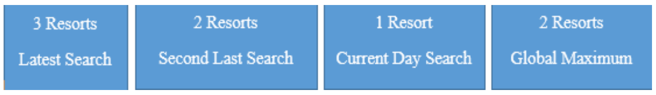

Fig 4. Recommendation distribution

B. *Model 2*

Model 2 can be decomposed into four parts, each explained below

a) Popularity Factor (P_i): For each resort i, P_i is nothing but the ratio of the number of times resort i was booked to the total number of confirmations.

b) Cosine Similarity (CS_{ij}): Cosine similarity, CS_{ij} tells us how likely a customer j is to book resort i based on resort amenities. CS_{ij} is calculated using the following steps:

   - Step 1: Calculate user vector u_j, which is the average of the resort amenities vector for the resort of each booking that the user has made.

   - Step 2: Calculate cosine similarity between each resort vector (r_i) and user vector (u_j), CS_{ij} with the following formula:

$$CS_{ij} = Cossim(r_i, u_j) = \frac{A.B}{|A||B|}$$

c) Search Factor (S_{ij}): If the user i has searched for a resort j before today, then the search factor of the resort for the user increases as follows:

$$Sij = Sij + \frac{\lambda}{T+1}$$

Where λ is the tuning parameter. In our model, we take it to be 1, and T is the difference in days between the search and today.

d) Current Search Factor (C_{ij}): If the user searches for a region, destination, city or resort, the current search factor for all resorts in that region, destination, city and for the resort increases by 0.01, 0.02, 0.03, and 0.04 respectively.

After calculating all these four features, we finally calculate the resort score for each resort i and user j as:

$$Score = P_i + \beta_1 * CS_{ij} + \beta_2 * S_{ij} + \beta_3 * C_{ij}$$

Where

β_1 = Number of confirmations of user j

β_2 = Number of searches of user j till the previous day

β_3 = Number of searches of user j today

After the resort score is calculated for each resort, the resorts are sorted by the resort score and the top eight resorts in the region searched are recommended to the user.

V. RESULTS

Fig. 5. compares the model accuracy of the two models implemented and the existing model. It is to be noted that the current context plays a crucial role in determining the booking behavior of a user.

Model 1, which uses matrix factorization gets an accuracy of 50% on the test dataset. Model 2, which uses cosine similarity is 65% accurate. This means that users on an average booked resort recommended to them in the last 17 days, 50% of the times when we use Model 1 and 65% when we use Model 2.

Table 4 compares the two models implemented. Due to the diversification involved in Model 1, Model 1 is a better model if one wants to promote under-booked hotels. Model 2, however captures the user search behavior well and can be used to provide relevant recommendations to the user.

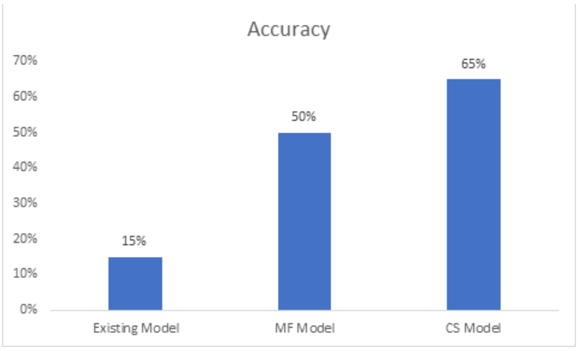

Fig 5. Model accuracy comparison

TABLE 4. SURGE TABLE

	Model 1	**Model 2**
Average Recommendation Time	1.5 seconds	1.13 seconds
Accuracy	50%	65%
Diversification	Recommendations diversified by search history, search frequency and historical bookings.	Recommendations confined to the current region searched for.
Factors Considered for Modeling	User-user similarity, resort-resort similarity	User search history, resort amenities, resort popularity

VI. CONCLUSION

A personalized recommendation system based on the user's historic data & current searches, influences customer's decision. It leads to a better customer experience, boosts sales, increase loyalty & today customers expect it. Most of the existing models are time-consuming, therefore there was a need for a more dynamic modelling approach to incorporate recent searches and come up with enhanced recommendations. Two models are developed to serve this purpose. Model 1 is based on matrix factorization, incorporating previous bookings, search weights added with decay function of time valuing recent searches and penalizing older searches, followed by machine learning ranking and filtering. This model gave significantly improved results and can be used for limited users. Its benefit is that it improves as more data inflows. Even though it takes time to run based on the size of the matrix, but that can be reduced based on the clustering of the users and the items. Matrix factorization also works even if the products are not similar.

The second model is based on the features of the resorts and the historical booking for users and their respective searches. It is based on the concept that a user tends to book similar resorts every time depending on the amenities. It can be further improved if the price and availability can be incorporated. This model gave better accuracy in comparison to the previous models. This model takes more time for individual recommendations but takes much lesser time to calculate user vector at the end of the day and can cater to numerous customers. It assumes that all the products that we are recommending have a similar feature. In addition to giving personalized recommendations to individual users, this model can also be used to suggest the change in amenities of the resorts that can increase their sales and eventually leading to higher revenue.

Both models can be used for the recommendation engine based on the requirement of the client and the products. A hybrid model of the two algorithms can be developed and may lead to better results. All the testing has been done based on historical data, and for their true accuracy, it can be done in the live environment using A/B testing or click to conversion ratio. Overall a novel idea has been recommended in this paper which recommends quickly as per the current search. Further research can be done based on the problem statement and data available..

REFERENCES

[1] Personalized Product Recommendations Drive Just 7% of Visits but 26% of Revenue. (n.d.). Retrieved from https://www.salesforce.com/blog/2017/11/personalized-product-recommendations-drive-just-7-visits-26-revenue.html

[2] How Netflix's Recommendations System Works. (n.d.). Retrieved from https://help.netflix.com/en/node/100639

[3] Keller, E. (2012, July 27). Recommendations are What Drives Your Business. Remember to Ask for Them. Retrieved from https://www.forbes.com/sites/kellerfaygroup/2012/07/25/recommendations-are-what-drives-your-business-remember-to-ask-for-them/#62fa5f9539c6

[4] Deng, H. (2019, December 5). Recommender Systems in Practice. Retrieved from https://towardsdatascience.com/recommender-systems-in-practice-cef9033bb23a

[5] Challenges & Solutions for Production Recommendation Systems – Data Revenue Blog. (n.d.). Retrieved from https://www.datarevenue.com/en-blog/building-a-production-ready-recommendation-system

[6] Shahabi, Cyrus & Banaei-Kashani, Farnoush & Chen, Yi-Shin & McLeod, Dennis. (2001). Yoda: An Accurate and Scalable Web-Based Recommendation System. 418-432. 10.1007/3-540-44751-2_31.

[7] Jin, Xin & Zhou, Yanzan & Mobasher, Bamshad. (2005). A maximum entropy web recommendation system: Combining collaborative and content features. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 612-617. 10.1145/1081870.1081945.

[8] Y. Hu, Y. Koren and C. Volinsky, "Collaborative Filtering for Implicit Feedback Datasets," 2008 Eighth IEEE International Conference on Data Mining, Pisa, 2008, pp. 263-272.

[9] G. Huming and L. Weili, "A Hotel Recommendation System Based on Collaborative Filtering and Rankboost Algorithm," 2010 Second International Conference on Multimedia and Information Technology, Kaifeng, 2010, pp. 317-320.

[10] Chandramouli, Badrish & Levandoski, Justin & Eldawy, Ahmed & Mokbel, Mohamed. (2011). StreamRec: a real-time recommender system. 1243-1246. 10.1145/1989323.1989465.

[11] D.A. Adeniyi, Z. Wei, Y. Yongquan,Automated web usage data mining and recommendation system using K-Nearest Neighbor (KNN) classification method,Applied Computing and Informatics,Volume 12, Issue 1,2016,Pages 90-108,ISSN 2210-8327,https://doi.org/10.1016/j.aci.2014.10.001.(http://www.sciencedirect.com/science/article/pii/S221083271400026X)

[12] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, Grammati Pantziou, Mobile recommender systems in tourism,Journal of Network and Computer Applications, Volume 39,2014,Pages 319-333,ISSN 1084 8045, https://doi.org/10.1016/j.jnca.2013.04.006.(http://www.sciencedirect.com/science/article/pii/S1084804513001094)

[13] Chen, T., Chuang, Y.H. Fuzzy and nonlinear programming approach for optimizing the performance of ubiquitous hotel recommendation. J Ambient Intell Human Comput 9, 275–284 (2018). https://doi.org/10.1007/s12652-015-0335-2

[14] Huang, Yanxiang, Bin Cui, Wenyu Zhang, Jie Jiang and Ying Xu. “TencentRec: Real-time Stream Recommendation in Practice.” SIGMOD '15 (2015).

[15] Lin, Y., Chen, T. & Wang, L. Integer nonlinear programming and optimized weighted-average approach for mobile hotel recommendation by considering travelers’ unknown preferences. Oper Res Int J 18, 625–643 (2018). https://doi.org/10.1007/s12351-016-0270-9

[16] Arruza, Michael, John Pericich and Michael Straka. “The Automated Travel Agent: Hotel Recommendations Using Machine Learning.” (2016).

[17] Yu-Cheng Lin & Toly Chen & Li-Chih Wang, 2018. "Integer nonlinear programming and optimized weighted-average approach for mobile hotel recommendation by considering travelers’ unknown preferences," Operational Research, Springer, vol. 18(3), pages 625-643, October.

[18] Mavalankar, Aditi & Gupta, Ajitesh & Gandotra, Chetan & Misra, Rishabh. (2019). Hotel Recommendation System. 10.13140/RG.2.2.27394.22728/1.

[19] Malte Ludewig and Dietmar Jannach. 2019. Learning to rank hotels for search and recommendation from session-based interaction logs and meta data. In Proceedings of the Workshop on ACM Recommender Systems Challenge (RecSys Challenge ’19). Association for Computing Machinery, New York, NY, USA, Article 5, 1–5. DOI:https://doi.org/10.1145/3359555.3359561