Offline Evaluation using Interactions to Decide Cross-selling Recommendations Algorithm for Online Food Delivery

Manchit Madan

Delivery Hero, Berlin, Germany

Abstract

With the rapid growth of online food delivery, offering personalized recommendations is crucial in reducing the choice paradox for users. This paper solves the problem of deciding the best cross-selling recommendations algorithm by introducing a Cart Offline Evaluation Framework (*COEF*) that caters to the business objectives with the help of targeted metrics. It further addresses the problem of creating the primary context (required for giving recommendations). A comparison of offline metrics with online metrics (using three A/B tests on a major food delivery platform) shows that our approach is able to predict the direction-wise (positive/negative) online performance based on the percentage improvement in the offline metrics. The winning algorithm is already deployed on the platform.

Keywords

Offline Evaluation, Recommender Systems, User Interactions, Data Augmentation, Custom Metrics

1. Introduction

The global revenue from the online food delivery market is expected to rise from $300 billion in 2022 to $450 billion by 2023 [1]. Enhancing user experience by personalizing recommendations has been one of the key reasons ([1], [2]). But, evaluating recommender systems is a challenging task. Broadly, there are two ways of evaluation: Online (A/B tests) & Offline. One can often offline evaluate multiple ideas in a short amount of time just by changing algorithms, feature sets, hyper-parameters, etc but what we can test online is always limited by time, the number of users on the platform, and the risk of negatively impacting the user experience which could further impact the business metrics. Thus, offline evaluation can help alleviate this by giving some indication of which algorithms are likely to perform well online.

This paper focuses on creating an offline evaluation framework, using the interactions data for a major online food delivery platform operating in various markets around the world. The framework is being leveraged to offline evaluate cross-sell recommender systems on the cart page for different business KPIs. Contrary to previous studies ([3], [4], [5]), our logs only have partial information (primary context is absent) stored due to design issues. Hence, the contributions are three-fold and are summarised as follows:

- A novel approach to generate an offline evaluation dataset using users' interaction with the given recommendations, in the absence of the primary context (items in the cart)
- Custom offline evaluation metrics that comply with the business objectives
- Comparison of offline evaluation with three A/B tests to test the efficacy of our Cart Offline Evaluation Framework (*COEF*)

2. Related Work

Instead of either a purely online (or offline) evaluation, one can reach the middle ground by offline evaluating using historical data collected online using logs generated from users' interaction with a recommender system, including "ground truth" inferred from interactions such as clicks, add to cart, remove from cart. Gunawardana *et al* emphasized the need to take into account the user's natural browsing behavior [6]. But it could be challenging to incorporate historical log data for offline evaluation due to various biases present: "trust bias" which leads to more clicks on links ranked highly by Google, "quality-of-context bias": clicking decision is also influenced by the overall quality of the other abstracts in the ranking [7], position bias [8], attractiveness bias [9].

There are some studies that have questioned the validity of offline evaluation and pointed out that error-based, and rank-based metrics computed on offline datasets do not correlate with online results ([10], [11], [12]). Gruson *et al*. presented a study showing offline evaluation results can predict online A/B test results under the right approach to de-biasing [4]. In this work, we have curated an offline evaluation dataset and metrics that help us predict the direction (positive or negative) of the online

Workshop on Learning and Evaluating Recommendations with Impressions (LERI) @ RecSys 2023, September 18-22 2023, Singapore
manchit.madan@deliveryhero.com (M. Madan)
https://www.linkedin.com/in/manchit-madan/ (M. Madan)
0000-0002-4730-4672 (M. Madan)

[1] https://www.statista.com/statistics/1170631/online-food-delivery-market-size-worldwide/

metrics.

Lastly, we discuss different offline evaluation datasets & metrics used. Perez *et al*. conducted an evaluation study of recommendation models for the traditional top-N recommendations using impressions data from open-sourced datasets. The evaluation was done against the test split [3]. Paraschakis *et al*. tried both non-chronological & chronological split of train-test data [5]. Agarwal *et al*. used the first product (that the user interacted with) as the context & remaining products are assigned to the ground truth set [13]. Talking about the metrics, Paraschakis *et al*. measured precision at the level of recall (R-precision) as it adjusts for the size of the customer's test set [5]. Other metrics such as Precision, Recall, Mean Average Precision, Novelty, Diversity[14] and Normalized Cumulative Discounted Gain ([15], [16], [17]) are commonly used in evaluating recommender systems. While for most of the approaches, either the context was fully available, or the exact context was not used, our approach aims to create the primary context (items in the cart) for cart cross-selling recommendations. Also, since the Average Basket Value is one of the top business objectives in this domain, we've introduced some new offline evaluation metrics.

3. Methodology

The problem statement involves using offline evaluation of cart cross-sell recommendations for a major online food delivery platform to decide the best algorithm for a successful online A/B test. One of the purposes of cart recommendations is to complete the cart, thereby helping users add products that complement their existing items in the cart. From a business point of view, the objective is to increase the Average Basket Value (ABV): (sum of orders' value/number of orders) and the Order Share (number of orders with products added from recommendations/total number of orders).

3.1. Cross-selling Recommenders

In this section, we give an overview of cross-selling algorithms that have been experimented on our platform. The algorithms accept four inputs (items in the cart, user features, vendor features, and time of the day) and return the candidate items (items present in the vendor's menu) sorted based on the score predicted by the model. Initially, a heuristics model M_{HEU} was built using the historical orders. The candidate items were ranked using a formula that multiplied 1) similarity of past orders with current cart, 2) number of items in the current cart that have been previously purchased by the user, and 3) mean time difference between the current cart & past orders.

Secondly, a CatBoost Ranker M_{CAT} [18] (gradient-boosted tree with pair-wise & group-wise losses, wherein grouping was done at *order id* level) was built using the historical orders data. The positive training data comprised of historical orders, where the items (P_1, P_2, P_3, P_4) in an order were split into cart (P_1, P_2, P_3) & positive candidate product P_4 (last product in an order, based on the timestamp of adding it to the cart). Since most of the orders had up to 4 items, only those orders were considered for training. Orders having fewer than 4 items were padded by a dummy item. For every *order id*, negative candidates were chosen randomly from the vendor's menu such that they don't match with the items in the cart & the positive candidate product. Hence, the model M_{CAT} was trained on binary labels. Categorical & numerical features of item, user & vendor were created, including pair-wise Lift of candidate item with every item in the cart (inspired by Market Basket Analysis [19]), co-bought frequency of candidate item with the cart (at overall, customer & vendor level), customer order history (cart size, item preferences, etc), price features, hour of the day features. Hyper-parameter optimization was done on the validation split (chronological split: last 1 week of training data).

Thirdly, this model was used to create $M_{Price\ CAT}$ that had multi-class labels (instead of binary labels) based on whether the item was purchased with the cart (1, 2, 3: using price of the candidate item, 3 is the highest price class) & not purchased with the cart (0). Also, the product names were preprocessed (lowercase, remove punctuations, etc) which removed 10% sparsity in the data. Finally, the last model $M_{Food\ Ont}$ was built on top of $M_{Price\ CAT}$ having additional features based on the food ontology (some prior studies using them:[20], [21]) & candidate selection of items (based on food ontology-based categories that have been purchased together in the last 6 months) with aim to improve relevance. Note that since there were no latency issues in the previous models (as the maximum number of items in a vendor's menu is less than 200), candidate selection was not done.

3.2. Dataset

Whenever a user lands on the cart page, item recommendations are presented based on the existing items in the cart, user preference, available menu items & time of the day. The dataset consists of impressions (recommendations) & users' interaction with these impressions. Figure 1 explains the series of events that are triggered based on the user's journey. The event "*recos.loaded*" is fired when recommendations are loaded. Interaction is defined in terms of Add to Carts (*ATC*) and Remove from Carts (*RFC*). The assumption is that these different ways in which a user interacts with an item indicate different intents. The aim is to evaluate the performance of the

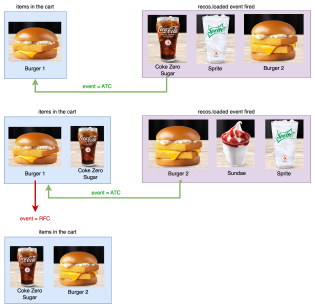

Figure 1: Events triggered when a user lands on the cart page

cross-selling recommenders & compare the recommendations with interactions data (*ATC*, *RFC*) that we consider as ground truth in this setting. But a major issue in the existing dataset is the absence of the "items in the cart" when certain recommendations were shown and users interacted with those recommendations.

3.3. Cart Offline Evaluation Framework

Our Cart Offline Evaluation Framework (*COEF*) can be divided into three components:

- Create primary context: Items in Cart
- Aggregate item interactions
- Combine context with interactions

3.3.1. Create primary context: Items in Cart

Given user sessions S, user u & vendor v, the steps for generating the primary context are presented in Algorithm 1. From S, S_1 is created by filtering for sessions having at least one *ATC* from cart recommendations. Since a user can interact with multiple vendors (restaurants) in a session, S_1 is filtered for sessions having maximum n vendors (decided based on D_v, distribution of number of vendors interacted at session level) to create S_2, thereby removing noise. Example from one market: based on D_v, $n = 3$ is chosen as it covers 90% sessions. $\forall\ S_2$, v, "recos.loaded" R_l events are sorted in the ascending order of their timestamps ts. To get the primary context (items in the cart) for every R_l, items I having *ATC*, *RFC* events are considered such that ts_{ATC}, ts_{RFC} <= ts of R_l. Hence, R_l has *ATC*, *RFC* from all R_{l-1}, R_{l-2},...,etc. Then for every R_l, the items in the cart $I*$ are those satisfying:

$$N_{ATC} - N_{RFC} > 0 \tag{1}$$

where N_{ATC}, N_{RFC} are the number of add to carts, remove from carts respectively. Items $I*$ are sorted based on their minimum ts_{ATC}. Since a non-logged-in user could log-in anytime during their session, statistical mode [22] of u is assumed at S_2, R_l level. Hence, the output of Algorithm 1 is R_l, u, v, items in the cart & ts of R_l.

Algorithm 1: Create Items in the Cart

Input:
- User sessions S,
- vendor v,
- user u,
- Events set E* = {R_l, *ATC*, *RFC*}

Output: R_l, u, v, items in the cart, ts of R_l

1 Sessions with >= 1 *ATC*, $S_1 \leftarrow S$;
2 Sessions with max n vendors, $S_2 \leftarrow S_1$;
3 $\forall\ S_2$, v ; ascending order sort R_l based on ts ;
4 $\forall\ R_l$, select items I s.t ts_{ATC}, ts_{RFC} <= ts of R_l ;
5 Select items $I*$ s.t N_{ATC} - N_{RFC} > 0 ;
6 Sort items $I*$ based on min. ts_{ATC} ;
7 $\forall\ S_2$, R_l take mode of u ;

3.3.2. Aggregate item interactions

Algorithm 2 describes the steps for aggregating the item interactions. Since we are only interested in user interactions after the user has reached the cart page, filter S_2 such that ts of *cart.clicked* event <= ts of *ATC*, *RFC*. This gives us S_3 (almost 96% of the sessions from S_2), from which we create episodes E_i by ranking *recos.loaded* R_l events in ascending order of their ts $\forall\ u$, v such that:

$$ts\,R_{l_i} <= ts\,E_i <= ts\,R_{l_i+1} \tag{2}$$

R_{l_i} gives the positions of items p_c shown to the user for every E_i. Since cart context is refreshed after every user interaction (*ATC* or *RFC*), *recos.loaded* is triggered & a fresh set of recommendations is given. To account for this, E_i has items with *RFC* in episode E_{i+1} as well. Then $\forall$ E_i, net *ATC* product interactions are given by equation 1. In order to estimate the relevance *rel* of each item at episode level, we assume that *ATC* holds higher weight than *RFC*, which in turn holds higher weight than no interaction. Items without interaction & those with both *ATC* & *RFC* get 0 weight, whereas items with *ATC* hold 3 times more weight than *RFC*. Details on these choices of weights are explained in the Results section 4.3.1. Note that the recommendation position is not considered while estimating the relevance. We aim to incorporate it in future works. The output of Algorithm 2 is E_i, R_{l_i}, u, v, item c, net *ATC*, N_{RFC}, rel_c, recommendation position p_c.

3.3.3. Combine context with interactions

Offline evaluation data D_{OE} is finally created by joining the outputs of Algorithm 1 & Algorithm 2 at episode E_i, "recos.loaded" R_{l_i}, user & vendor level to give the sets of impressions (recommendations) & interactions (user's response). Item's price information is also added & is used to create *Cart Value* by summing over the prices of

Algorithm 2: Aggregate product interactions

Input:
- Filtered sessions S_2,
- vendor v,
- user u,
- Events set E* = {R_l, ATC, RFC, $cart.clicked$}

Output: R_l, u, v, c, net ATC, N_{RFC}, p_c

1 Filter events s.t $ts_{cart.clicked}$ <= ts_{ATC}, ts_{RFC}, $S_3 \leftarrow S_2$;
2 $\forall$ u, v; create episodes E_i s.t. ts R_{l_i} <= ts E_i <= ts R_{l_i+1};
3 $\forall$ E_i; get recommendation position p_c from R_{l_i};
4 Context switch adjustment: E_i has items with RFC from E_{i+1} as well;
5 $\forall$ E_i, net ATC s.t N_{ATC} - N_{RFC} > 0;

all items in the cart. Though we cannot share the dataset due to legal constraints, the work presented in this paper can be replicated using algorithm 1, 2.

4. Results and Analysis

4.1. Analysis

Analysis shows that 39% sessions contain just a single episode, & 75% sessions contain <= 3 episodes. Looking at the importance of recommendation position: 35% times items are ATC from the 1st position, & top 3 positions account for 72% ATC, indicating a strong position bias. 37% times only a single item was present in the cart, 2 items were present 30% times, & 3 products 16% times. The price distribution of top 10 recommendation positions show an increasing trend: items added/recommended at 1st position have the lowest price & those at position 10 have the highest price.

Substitution happens whenever a user adds an item from recommendations to the cart & removes an item existing in the cart. It could lead to either an up-sell (if total price of ATC > RFC) or a down-sell (if total price of ATC < RFC). Assuming a cart is complete, there are higher chances of up-sell if the recommended items have the same category as the items in the cart (Figure 1 shows an up-sell since "Burger 1" was substituted by a more expensive "Burger 2"). Whereas, there are more chances of a down-sell if the recommended items have different categories as compared to the items in the cart.

4.2. Evaluation Metrics

Custom metrics are created to estimate the impact on business objectives (Average Basket Value (ABV) & Order Share). Let $value_{ATC}$ be the sum of prices of items added to cart, likewise $value_{RFC}$ be for items removed from cart.

The impact on ABV could be estimated by the following metrics (based on top k recommendations):

1. cart value impact (cvi) @k: average percentage improvement in the cart value after relevant items (recommended items present in the ground truth) are added, removed from cart. It is given by-

$$\frac{\sum_{i=1}^{k}(\frac{\text{new cart value - cart value}}{\text{cart value}})}{k} \tag{3}$$

where new cart value = cart value + $value_{ATC}$ - $value_{RFC}$

2. upsell @k: average number of up-sells, where upsell is a binary value based on the following equation-

$$\text{upsell @k} = 1; \text{ if } \sum_{i=1}^{k} value_{ATC} > \sum_{i=1}^{k} value_{RFC}; \text{ else } 0 \tag{4}$$

3. downsell @k: average number of down-sells at k^{th} threshold, given by-

$$\text{downsell @k} = 1; \text{ if } \sum_{i=1}^{k} value_{ATC} < \sum_{i=1}^{k} value_{RFC}; \text{ else } 0 \tag{5}$$

The impact on Order Share could be estimated by:

4. mean weighted avg. precision @k: mean of weighted average precision (wap), given by-

$$\text{wap @k} = \sum_{k=0}^{n-1}(R_k - R_{k+1}) * P_k * rel_i \tag{6}$$

where R_k, P_k are the precision, recall at the k^{th} threshold & rel_i is the relevance given to i^{th} item.

5. ndcg @k: Normalized Discounted Cumulative Gain, a standard information retrieval measure used for evaluating the goodness of ranking a set of items.

4.3. Results

Using the Cart Offline Evaluation Framework $COEF$ (Section 3.3), the offline evaluation dataset is prepared (example for 1 market, there are 186K episodes (samples), 134k users, 19k vendors. Using the context (items in the cart, user, vendor, time of the day) from the episodes, available items in the vendor's menu are ranked (in descending order of their probability of being ordered) using the Cart Recommender. All available items in the vendor's menu are ranked to mimic the actual online scenario. Then, for each episode, the recommendations are compared with ground truth interactions to come up with the offline evaluation metrics set {$COEF_m$} (Section 4.2).

4.3.1. Impact of Relevance Weights

In this section, we discuss the impact of the relevance weights of user interactions ATC, RFC on {$COEF_m$}. With an increase in the absolute value of relevance weights

	Offline Metrics					Online Metrics	
Experiment	**ndcg @3**	**mwap @3**	**cvi @3**	**upsell @3**	**downsell @3**	**ABV**	**Order Share**
M_{HEU} v/s M_{CAT}	23%	25%	16%	25%	14%	1% *	8% *
M_{CAT} v/s $M_{Price\ CAT}$	2%	2.4%	2.8%	2.1%	-1%	0.5% *	0.1%
$M_{Price\ CAT}$ v/s $M_{Food\ Ont}$	-1.2%	-1.5%	1.3%	1.5%	1.2%	-0.1%	0.3%

Table 1: Comparison of Offline metrics with Online metrics, * represents statistical significance

W, mwap @k & ndcg @k increases, whereas there is no effect on the ABV-related metrics (cart value impact (cvi) @k, upsell @k, downsell @k) since the relevance weights do not influence these metrics (Figure 2). Finally, the relevance weight of ATC (W_{ATC} = 2) and RFC (W_{RFC} = 0.7) were chosen based on the directional alignment of online metrics & offline metrics.

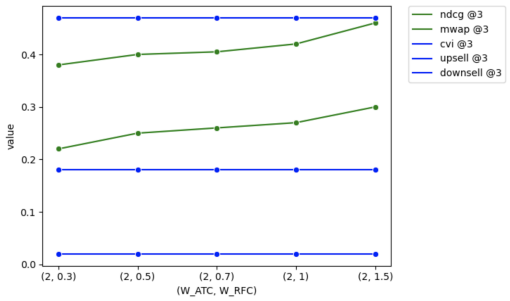

Figure 2: Impact of Relevance Weights on Offline metrics

4.3.2. Comparison of Offline v/s Online metrics

In this section, the live experiments (A/B tests) performed on one of the major online food delivery platforms are discussed. Experiments were conducted to compare the Cart Cross-sell Recommenders (introduced in Section 3.1). The first experiment was conducted to compare heuristics model M_{HEU} (Control) with ML model M_{CAT} (Variation). It was conducted in 5 markets (countries) & the experiment timeframe was decided based on the sample size estimation [23]. The primary (secondary) objective was to increase the Order Share (ABV). The overall online metrics showed that M_{CAT} performed better than M_{HEU} (1% increase in ABV, 8% increase in Order Share & both statistically significant with 95% confidence). The overall offline metrics $\{COEF_m\}$ showed that both Order Share-related metrics & ABV-related metrics improved. Table 1 compares offline metrics with online metrics for different experiments. Offline ABV metrics improved by 16% & Order Share metrics improved by 23%. The winning model M_{CAT} was rolled out to all the markets & was live for 6 months.

The next experiment compared M_{CAT} (Control) with its successor $M_{Price\ CAT}$ (Variation). This test was conducted in 8 markets, some of which were different than the previous A/B test. The offline metrics predicted a 2-3% increase in both ABV & Order Share related metrics. The experiment results showed a 0.5% statistically significant increase in ABV & a 0.1% non-significant increase in Order Share (see Table 1). Since, here the primary objective was to increase ABV (due to a shift in business priorities), the model $M_{Price\ CAT}$ was rolled out to all the markets & has been live for 9 weeks now.

The final experiment compared $M_{Price\ CAT}$ (Control) with its successor $M_{Food\ Ont}$ (Variation). The primary objective was again to increase ABV & experiment was run in 5 markets, some were again different from the previous A/B test. The offline metrics showed only a 1.3% increase in ABV-related metrics & a 1.2% decrease in the Order Share-related metrics. Online metrics showed that Variation decreased ABV by 0.1% & improved Order Share by 0.3%, but both of these were not statistically significant(see Table 1). Hence, the Variation ($M_{Food\ Ont}$) was not rolled out & would need more improvements.

The above experiments show that our $COEF$ is able to confidently predict the direction-wise (position/negative) online performance with statistical significance if the increase in offline metrics is significantly high. The experiments were conducted in different markets, making them robust to the properties of markets. Also, since the timeframe used to create the offline evaluation dataset is after the launch of M_{CAT}, $\{COEF_m\}$ gives the same direction as the online metrics since the baseline algorithm (CatBoost) is the same. These results would be further refined based on future experiments.

5. Conclusion

In this paper, we solve the challenging problem of creating an accurate offline evaluation dataset for offline evaluating cart cross-sell recommendations & show its effectiveness by comparing the offline metrics with the online metrics. Our Cart Offline Evaluation Framework ($COEF$) creates the primary context using the ATC & RFC events based on the users' interactions and uses custom metrics to estimate the directional impact on business metrics. As future work, we plan to improve our $COEF$ by considering the position bias present in the data.

References

[1] Y. Wang, L. Tao, X. Zhang, Recommending for a three-sided food delivery marketplace: A multi-objective hierarchical approach, Technical Report, Working paper, Google Brain, Mountain View, CA, 2022.

[2] C. N. Sánchez, J. Domínguez-Soberanes, A. Arreola, M. Graff, Recommendation system for a delivery food application based on number of orders, Applied Sciences 13 (2023) 2299.

[3] F. B. Perez Maurera, M. Ferrari Dacrema, P. Cremonesi, Towards the evaluation of recommender systems with impressions, in: Proceedings of the 16th ACM Conference on Recommender Systems, 2022, pp. 610–615.

[4] A. Gruson, P. Chandar, C. Charbuillet, J. McInerney, S. Hansen, D. Tardieu, B. Carterette, Offline evaluation to make decisions about playlistrecommendation algorithms, in: Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, 2019, pp. 420–428.

[5] D. Paraschakis, B. J. Nilsson, J. Holländer, Comparative evaluation of top-n recommenders in e-commerce: An industrial perspective, in: 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), IEEE, 2015, pp. 1024–1031.

[6] A. Gunawardana, G. Shani, S. Yogev, Evaluating recommender systems, in: Recommender systems handbook, Springer, 2012, pp. 547–601.

[7] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, G. Gay, Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search, ACM Transactions on Information Systems (TOIS) 25 (2007) 7–es.

[8] N. Craswell, O. Zoeter, M. Taylor, B. Ramsey, An experimental comparison of click position-bias models, in: Proceedings of the 2008 international conference on web search and data mining, 2008, pp. 87–94.

[9] Y. Yue, R. Patel, H. Roehrig, Beyond position bias: Examining result attractiveness as a source of presentation bias in clickthrough data, in: Proceedings of the 19th international conference on World wide web, 2010, pp. 1011–1018.

[10] M. Rossetti, F. Stella, M. Zanker, Contrasting offline and online results when evaluating recommendation algorithms, in: Proceedings of the 10th ACM conference on recommender systems, 2016, pp. 31–34.

[11] J. Beel, S. Langer, A comparison of offline evaluations, online evaluations, and user studies in the context of research-paper recommender systems, in: Research and Advanced Technology for Digital Libraries: 19th International Conference on Theory and Practice of Digital Libraries, TPDL 2015, Poznań, Poland, September 14-18, 2015, Proceedings 19, Springer, 2015, pp. 153–168.

[12] G. G. Gebremeskel, A. P. de Vries, Recommender systems evaluations: Offline, online, time and a/a test (2016).

[13] P. Agarwal, S. Vempati, S. Borar, Personalizing similar product recommendations in fashion e-commerce, arXiv preprint arXiv:1806.11371 (2018).

[14] M. D. Ekstrand, F. M. Harper, M. C. Willemsen, J. A. Konstan, User perception of differences in recommender algorithms, in: Proceedings of the 8th ACM Conference on Recommender systems, 2014, pp. 161–168.

[15] K. Järvelin, J. Kekäläinen, Cumulated gain-based evaluation of ir techniques, ACM Transactions on Information Systems (TOIS) 20 (2002) 422–446.

[16] P. Cremonesi, Y. Koren, R. Turrin, Performance of recommender algorithms on top-n recommendation tasks, in: Proceedings of the fourth ACM conference on Recommender systems, 2010, pp. 39–46.

[17] M. Madan, A. Chouragade, S. Vempati, The joy of dressing is an art: Outfit generation using self-attention bi-lstm, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2021, pp. 218–233.

[18] A. V. Dorogush, V. Ershov, A. Gulin, Catboost: gradient boosting with categorical features support, arXiv preprint arXiv:1810.11363 (2018).

[19] M. Kaur, S. Kang, Market basket analysis: Identify the changing trends of market data using association rule mining, Procedia computer science 85 (2016) 78–85.

[20] M. El-Dosuky, M. Z. Rashad, T. Hamza, A. El-Bassiouny, Food recommendation using ontology and heuristics, in: Advanced Machine Learning Technologies and Applications: First International Conference, AMLTA 2012, Cairo, Egypt, December 8-10, 2012. Proceedings 1, Springer, 2012, pp. 423–429.

[21] T. N. Trang Tran, M. Atas, A. Felfernig, M. Stettinger, An overview of recommender systems in the healthy food domain, Journal of Intelligent Information Systems 50 (2018) 501–526.

[22] T. Dalenius, The mode–a neglected statistical parameter, Journal of the Royal Statistical Society. Series A (General) (1965) 110–117.

[23] R. Kohavi, R. Longbotham, D. Sommerfield, R. M. Henne, Controlled experiments on the web: survey and practical guide, Data mining and knowledge discovery 18 (2009) 140–181.