Personalizing Item Recommendation via Price Understanding

Soumya Wadhwa
soumya.wadhwa@walmartlabs.com
Walmart Labs, Sunnyvale, California

Ashish Ranjan
ashish.ranjan@walmartlabs.com
Walmart Labs, Sunnyvale, California

Selene Xu
yue.xu@walmartlabs.com
Walmart Labs, Sunnyvale, California

Jason H.D. Cho
hcho@walmartlabs.com
Walmart Labs, Sunnyvale, California

Sushant Kumar
skumar4@walmartlabs.com
Walmart Labs, Sunnyvale, California

Kannan Achan
kachan@walmartlabs.com
Walmart Labs, Sunnyvale, California

ABSTRACT

Personalization has gained a lot of traction in the e-commerce domain since there is ample evidence for short-term and long-term benefits of understanding user preferences and ensuring user satisfaction. However, effectively personalizing recommendations is a challenging task, especially at scale. Price is often a key consideration for purchases, and user behavior varies widely depending on demographic and psychological factors. While difficult to model, this is an important signal to consider for user-item recommendation. In this paper, we focus on personalizing and improving the relevance of item recommendations for e-commerce users by leveraging price as an essential input. More concretely, we segregate items into price bands indicating how expensive they are, infer user affinity to price bands based on historical behavior and use features derived from this knowledge to re-rank items in a real-world recommendation scenario. We experiment with various statistical and machine learning methods to determine item price bands, user price affinities and item price similarities, and demonstrate impact on the recommendation quality for millions of users and items.

1 INTRODUCTION

Recommender systems are ubiquitous on websites today. Recommendation algorithms can be based on item-item interactions or user-item feedback. In recent times, websites are increasingly focusing on providing an experience tailored to their users [35] [10] [36] with personalization at the segment or individual level. Understanding user preferences and recommending relevant items to them accordingly has been shown to improve user satisfaction and conversion rates, which is a win-win situation [4] [5]. While it is essential, scaling the personalization of recommendations anchored on combinations of users and items is very challenging, especially in the e-commerce domain where millions of users can potentially interact with millions of items.

For most users, price is often a key factor for making purchases [14] [8] [34]. Users make price-value trade-offs when they purchase products, and their behavior can vary widely depending on demographic factors such as their salary or location and psychological factors such as money consciousness or additional interest in certain types of products. Let us consider two users. The first user is a sound engineer and is looking to purchase high-quality headphones for use at work. Since this user needs to discern any imperfections, they may be looking to purchase expensive headphones. Another user may decide to purchase headphones to listen to podcasts. As long as this user can understand the podcast, sound quality is not an issue and they can buy lower-priced headphones. To effectively

personalize their shopping journey, understanding that the first customer is looking for higher priced headphones and the second customer is looking for lower priced ones will help recommend products they are looking for.

However, defining what constitutes a high-priced or low-priced item is a difficult task. In the e-commerce domain, products, of course, have price associated with them. But, we do not know whether a given price is considered expensive or inexpensive for a given type of product, for example, a light bulb and a laptop. $100 may be a bargain for the laptop, but the same price-tag for the light bulb might make it very expensive. Similarly, we need to understand item prices for each product type and categorize them into different price bands (e.g. low vs. high) based on this. Subsequently, we can start understanding which price bands users are likely to purchase from for different product types.

To summarize, using price to personalize item recommendation is challenging because user price preferences need to be implicitly inferred and vary based on the type of product. Additionally, item prices are not sufficient to determine if a product is considered expensive versus not, and need to be standardized such that they can be compared across different types of products. In this paper, we aim to model user price affinity and item price similarity, and utilize them as input signals along with item-item relevance scores to personalize and improve the quality of item recommendations for e-commerce users. We achieve this using the following:

- Unsupervised methods to divide items into price bands indicating their degree of expensiveness
- Supervised methods to compute user affinity to different price bands based on their historical interactions
- Item and user price-related features to re-rank items in an actual user-item recommendation setting

This is done at the *Product Type* (PT) level which is the most granular level of the product taxonomy available in the Walmart product catalog. We use a large e-commerce dataset, and experiment with multiple statistical and machine learning methods to determine item price bands, user price affinities and item price similarities. We quantitatively show the positive impact on recommendation quality upon including price-related features in the re-ranking algorithm.

2 RELATED WORK

There has been extensive research on recommender systems and personalization. Many research efforts have been focused on collaborative filtering-based techniques. Traditional matrix factorization (MF) models [24] and variants [17] [31], incorporating implicit feedback, temporal effects and confidence levels, have proved superior to the classic nearest neighbor approaches in recommending items

[33]. Factorization machines [30] have also been used for recommendation to overcome feature sparsity issues. Emerging deep learning based solutions [6] [15] [13] have also shown promising results for recommendation. Item embeddings can be used to compute item-item similarity and recommend items accordingly [37] [11]. For modeling recommendations based on short session-based data, a sequential Recurrent Neural Network [28] (RNN)-based approach can be used to predict the next item [16]. More recently, causal embeddings for recommendation [3] have shown significant improvements over state-of-the-art factorization methods.

Price is an important factor to consider for users while making an online purchase. In [34], a conceptual framework is developed to explain the effects of the online medium on customer price sensitivity. User price sensitivity and price thresholds are discussed in [14]. Traditional and online supermarkets are compared in terms of user behavior with respect to brand, price and other search attributes in [8], and price sensitivity is found to be higher online. There are also other studies about impact of advertising [20] and brand credibility [9] on price sensitivity. In [38], the potential effect of the consumption occasion (functional vs. hedonic), social context and household income on users' price sensitivity is analyzed. There is substantial additional literature on consumer price sensitivity.

However, price has received relatively less attention as an input signal for recommendation. Price is used as a feature in [1] for personalization in the e-commerce domain by taking the ratio of the price of the current item to the average price of previously clicked items. There is also a brief mention of how price can affect user's affinity towards an item in [17]. Authors in [18] perform data analysis of logs to investigate what makes recommendations effective in practice, and include some factors based on "price levels" per product category. However, methods for determining these price levels are not discussed and user price affinity is incorporated as an average of recent price levels (not per category). Their focus is on the impact of popularity, discounts, reminders and recency on user click behavior. In [12], Willingness To Pay (WTP) distributions per user and product are modeled, and this is used with discount indication and seller reputation in a context-aware recommendation model to improve recommendation quality. More recently, [40] model the transitive relationship between user-to-item and item-to-price using Graph Convolution Networks [22] (GCN) to make the learned user representations price-aware. They incorporate prices and categories as nodes along with users and items in a heterogeneous graph. They also consider price as a categorical variable and discretize the price value into separate levels based on price ranges but do not experiment with different methods for this.

In our work, we explore several methods to compute item price bands and explicitly model user price affinity for various types of products, such that these input signals can be leveraged generally for use cases such as recommendation and search. We demonstrate results for user-item and item-item price-related features obtained from different model variations used together with an item-item relevance score for personalizing item-anchored recommendations.

3 METHODOLOGY

Our goal is to use past item interaction data (such as clicks and add to carts) for a given user and predict their affinity for a particular price band, and eventually incorporate this price understanding into recommendations. To achieve this, items are clustered into price bands at the product type level (Section 3.1), and then user activity patterns are learnt with respect to these item price bands to predict the probability that the user will purchase an item from a particular price band versus others for that product type. These predicted user-item price band affinity scores (Section 3.2) and item-item price band similarity scores (Section 3.3) are used as features along with relevance scores to re-rank item-anchored recommendations (Section 3.4) for personalization using price understanding.

3.1 Item Price Bands

Item prices vary a lot, from less than 10 dollars for a USB drive to thousands of dollars for a QLED television. However, to decide if an item is expensive or not, just the absolute value of price is insufficient. It is also important to take into consideration the product type since a price of $100 might be low for televisions, but high for a USB drive. Thus, we need to create representations for prices of items for each product type such that they are directly comparable across different items. So, we assign each item to one out of n bands using unsupervised methods since labels are unavailable.

3.1.1 Statistical Methods. We first explore statistical methods using item prices for each product type.

- *Range-Based*: For example, say television prices vary from $100 to $5100, and we decide to create 3 price bands with price range ratios 3:5:2. Then one unit of the range becomes ($5100 - $100)/(3+5+2) = $500, and the lowest price band extends from $100 to $100 + 3*$500 (= $1600), the middle from $1600 to $2600 + 5*$500 (= $4100) and the highest from $4100 to $4100 + 2*$500 (= $5100).
- *Percentile-Based*: For example, say, in the above situation, we decide to create 3 price bands with percentiles 30%, 50% and 20%. If there are 200 televisions on our item catalog with 60 TVs having price less than $500, 100 TVs with prices between $500 - $2500 and 40 TVs with prices greater than $2500, then those delineate the price bands.

Splitting items into equal bins based on range or percentiles did not work well in practice due to skew in item price distributions and transaction volumes. Creating unequal bins needs extensive manual tuning.

3.1.2 Clustering. Next, we use some common clustering methods to automatically put items from each product type into n clusters using the item price values.

- *K-Means* [26]: Each item is assigned to the cluster for which the mean price value is closest to the item price.
- *Gaussian Mixture Model (GMM)* [32]: We assume that all the price values are generated from a mixture of a finite number of normal distributions with unknown means and variances (estimated using Expectation Maximization). We pick the highest probability cluster as the item's price band.

3.1.3 Transaction Balancing. Another method is based on computing cumulative transaction volumes after arranging items in increasing order of their price value, and determining price band boundaries such that each price band accounts for an equal volume of item transactions. This technique was devised to mitigate data

imbalance in the subsequent step of using price bands to compute price affinities based on user activity.

3.2 User Price Affinity

After assigning price bands to items at the product type level, the next step is to determine the user price affinity per product type. In other words, given a product type, we want to predict the probability that the user will purchase an item from a certain price band versus others. For example, say we are considering two price bands, "expensive" and "cheap". Sam might have affinities 0.8 and 0.2 towards expensive and cheap fitness trackers, but 0.3 and 0.7 towards expensive and cheap bed frames. This indicates that she likes buying expensive fitness trackers but inexpensive bed frames. To predict this, we use historical data to train various machine learning models, using 6 months' data for generating features and the next 1 month for labels. The baseline prediction is based on the transactions in 6 months.

3.2.1 Baseline. For each user, per product type, we take the number of transactions (trx) in each price band (pb_i) and normalize it by summing transactions across all price bands for that product type (pt_j) and user to obtain affinity scores.

$$\text{user\_price\_affinity}_{(pb_i, pt_j)} = \frac{\text{\# of trx in } pb_i, pt_j}{\text{\# of trx in } pt_j} \quad (1)$$

3.2.2 Machine Learning. We consider user price affinity prediction as a multi-class classification (supervised machine learning) problem with number of classes equal to the number of price bands (n). For each user and product type, we use features such as number of transactions, add to carts and views of items per price band per month (for 6 months) by that user for that product type. We did not have ground truth data for labels. As a proxy, we use the price band which has the maximum number of transactions by the user for that product type, as the label. The aggregation of data for labels is done using the month following the last month used for feature generation. Thus, each data point is used to predict price affinities of a specific user towards different price bands in a particular product type. Subsequently, we use these features and labels to train and test multi-class *Logistic Regression* (LR) [23] and *Decision Tree* (DT) [25] models.

In LR, for each input data point (feature vector x and label y), the model learns weights (weight vector w) and outputs a probability distribution over the price bands (pb) for a given user and product type (pt) which represents their affinity.

$$\text{user\_price\_affinity}_{(pb_i, pt_j)} = \frac{\exp w_i^T x}{\sum_{k=1}^{n} \exp w_k^T x} \quad (2)$$

We use two variants - unweighted (LR-unbal) which is the vanilla model and weighted (LR-bal) based on class imbalance, where each data point is assigned a weight while contributing to the loss/gradient computation. The weight balancing heuristic used [21] is inversely proportional to class frequencies: $\frac{\text{n\_samples}}{\text{n\_classes} * \text{count\_y}}$, where y is the class label.

In DT, data is continuously split based on a certain feature at each step. We also consider random forests and fit multiple decision trees

on a number of smaller samples from the data. The final output is the average of different tree outputs.

3.3 Item Price Similarity

Another input is the similarity between price bands for items across product types based on user transaction patterns. For example, users who purchase medium-priced televisions might be likely to purchase high-priced sound bars and these (pt, pb) pairs are similar. This is also used as a feature while re-ranking to capture the item-item price similarity.

Pearson Correlation [2]: We compute the Pearson correlation (ρ) between observed transactions for each user for different (product type, price band) pairs (say pt_a, pb_i and pt_b, pb_j), and these are used as the price similarity scores.

$$\text{price2price} = \rho_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X * \sigma_Y} \quad (3)$$

Matrix Factorization [24]: We learn latent representations for (product type, price band) pairs by creating a user-(product type, price band) transaction matrix and factorizing it. $U^{m \times d}$ denotes the user representation (m users) and $V^{n \times d}$ denotes the (product type, price band) representation (n pairs). Embeddings are learned such that UV^T is a good approximation of transaction matrix T. Cosine similarity between these low-dimensional vectors are used as price similarity scores.

$$\text{price2price} = cos(\theta) = \frac{u \cdot v}{||u|| \, ||v||} \quad (4)$$

3.4 Re-Ranking

To tie everything up, we have a re-ranker engine that is capable of incorporating user price understanding and item price similarity into any item recommendation set such as Viewed also Viewed and Bought also Bought. We use an inference function to combine features related to user preference and item relevance, and predict user-item interactions:

$$P(u \text{ interacts with } r \mid u \text{ just interacted with } i) = f(g(u, r), h(i, r)) \quad (5)$$

where u is the user, i is the anchor item, r is the recommended item, $g(u, r)$ represents u's preference for r, and $h(i, r)$ represents item relevance between i and r. Currently, the inference function we use is simple logistic regression where user preference score and relevance score are combined linearly. The weights can be learned either at the global level (i.e. same weights across all product types) or at the product type level. There are more details in Section 4.4.

4 EXPERIMENTS AND RESULTS

We use a real-world proprietary e-commerce dataset from *walmart.com* for demonstrating results. We determine price bands for few million items and predict price affinity scores for millions of users across around 6000 product types.

4.1 Item Price Bands

We explored the trade-off between granularity for more useful user affinity scores and data sparsity issues in user-price band interactions as the number of price bands per product type increases, and decided to use $n = 5$ item price bands for our experiments. We

evaluate the discovered item price bands qualitatively, since ground truth labels are unavailable. One way is to look at how price ranges for different products were being split based on different methods described in Section 3.1. Of these, (as shown in Figure 1) we pick k-means and transaction balancing (transac-bal) as methods to further evaluate in the subsequent steps of predicting user affinities and using these to re-rank recommendations. We observe that clustering methods such as k-means put fewer very high priced items into the higher price bands, whereas trying to equalize number of transactions puts fewer items into the lower price bands, which is expected since less expensive items usually have more transactions. We also randomly sample items and inspect the quality of price bands. An example of televisions from 5 price bands (v low-0, low-1, medium-2, high-3 and v high-4) is shown in Figure 2.

4.2 User Price Affinity

We hold out 20% of data to test the trained user price affinity models described in Section 3.2. We use precision, recall and F1 score, which are common multi-class classification evaluation metrics, to assess the performance of different models. Since the classes in the data are not balanced, accuracy is not a good metric. Additionally, we also use the Mean Reciprocal Rank (MRR) to check whether even if the max transaction price band (ground truth label) does not get the maximum price affinity score, it gets a reasonably low (better) rank. Results for different models when price bands are determined using k-means and transac-bal are shown in Table 1 and Table 2 respectively. Random forests did not give much improvement over simple decision trees, so we have omitted those results. For the baseline, we obtain an overall MRR of around 0.51 for k-means and around 0.37 for transac-bal. All the machine learning methods performed better than the baseline. For logistic regression, we explored hyperparameters *aggregation depth:* [2,4], *maximum iterations:* [100,1000], *regularization:* [0,0.01] and *elastic net weights:* [0.4,0.8], and were able to obtain an overall MRR of around 0.85 for k-means and around 0.79 for transac-bal. For decision trees, we explored hyperparameters *impurity:* ["entropy", "gini"], *maximum depth:* [10, 20, 30] and *maximum bins:* [16, 32, 64], and were able to obtain an overall MRR of around 0.85 for k-means and around 0.76 for transac-bal. We observe that weighted / class-balanced logistic regression performs the best for both item price banding strategies, but performance varies across different price bands as seen in Figure 3, with metrics falling for higher price bands in the k-means case and remaining at similar levels in the transac-bal case.

4.3 Item Price Similarity

Figure 4 shows an example of results obtained for product type, price band pairs which are most similar to Medium-2 priced Bed Sheets. We leave quantitative evaluation of methods to the downstream re-ranking application.

4.4 Re-Ranking

We show how price understanding models perform when implemented on the "customers who viewed also viewed" (VAV) application (example shown in Figure 5). We take few million anchor items from VAV and limit to $N <= 30$ recommendations for each item ranked by a "relevance" score. This relevance score is based

on item-item features such as number of co-views, title match and popularity. The price understanding model offers two additional features for re-ranking on top of the relevance score: user price affinity score and price2price similarity score.

We first test out various methods used to develop user price affinity. We start with two main methods for item price banding: k-means and transaction balancing. For each item price banding model we have four variations for user price affinity: baseline, logistic regression (unbalanced), logistic regression (balanced) and decision trees. This gives us a total of eight versions of user price affinity scores. The inference function for re-ranking is balanced logistic regression:

$$y = w_0 + w_1 \times \text{relevance} + w_2 \times \text{user\_price\_affinity} \quad (6)$$

The weights in the equation above are trained at the global level (as opposed to at each product-type level) and are optimized for items that are co-viewed within each user session. To evaluate performance, we use common ranking evaluation metrics Normalized Discounted Cumulative Gain (NDCG) [19], Mean Hit Rate (MHR), Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP) [27] [7]. The offline evaluation results are shown in Table 4. We limit the evaluation metrics to be based on the top 5 recommendations. We observe that all the models outperform the relevance only model (no re-ranking). The best performing model uses transaction balancing for price banding and applies weighted logistic regression (balanced) to derive user price affinity scores.

We now expand on the previously established best performing model and supplement it with item price similarity information between the anchor item and the recommended item. We have two variations of item similarity scores to compare: Pearson correlation and matrix factorization. The inference function now has an additional feature, as follows:

$$
\begin{aligned}
y = w_0 &+ w_1 \times \text{relevance} \\
&+ w_2 \times \text{user\_price\_affinity} \\
&+ w_3 \times \text{price2price\_similarity}
\end{aligned}
\quad (7)
$$

Again we adopt a balanced logistic regression model to train the above objective function and learn the weights at the global level. The results are shown in Table 5. We observe an even greater boost in performance by adding the price2price feature. Overall, the best performing model uses price2price scores derived from matrix factorization. Compared to the relevance only model, this method shows 0.64% improvement in NDCG, 1% improvement in MHR, and 0.93% improvement in MAP. The improvements in NDCG, MHR and MAP@5 are statistically significant at 5% level in our offline evaluation. Though the MRR is slightly lower, the difference is not statistically significant. Also, since 5-6 recommended items are typically shown on the first pane of the module, metrics such as MHR become more important.

We further study the weights, w_1 w_2 w_3 from the inference function to gauge feature importance. After adjusting for feature variance (standard scaling of features), the ratio among the weights $w_1 : w_2 : w_3 = 33:3:1$. This tells us that the relevance score from the VAV model contributes the most even during re-ranking, but price-related features also add value. The user price affinity feature has greater weight than the price2price feature.
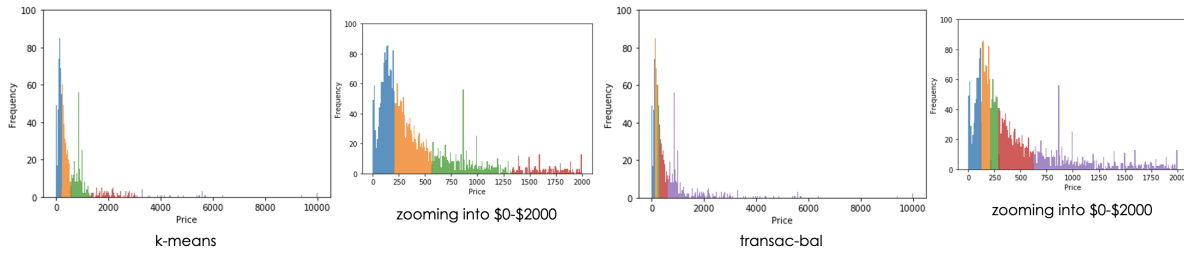
Figure 1: Different price bands obtained for k-means and transac-bal for product type Televisions

Televisions	Very Low	Low	Medium	High	Very High
Item	Refurbished Sharp 32" Class HD (720P) Smart LED TV (LC-32Q5200U)	Refurbished ELEMENT 50" Class FHD Smart LED TV (E2SW5018R)	SAMSUNG 43" Class 4K UHD 2160p LED Smart TV with HDR UN43NU6900	LG 60" Class 4K UHD 2160p LED Smart TV With HDR 60UM6900PUA	SAMSUNG 75" Class 4K Ultra HD (2160P) HDR Smart QLED TV QN75Q70R
Price	$99.99	$162.25	$282.49	$457.58	$2096.21

Figure 2: An example of items in different price bands obtained for product type Televisions

Method	Pr Band	Prec	Rec	F1	MRR
Baseline	V Low	0.51	0.56	0.53	0.77
	Low	0.36	0.33	0.34	0.64
	Medium	0.11	0.09	0.10	0.40
	High	0.02	0.02	0.02	0.26
	V High	0.002	0.001	0.001	0.20
LR-unbal	V Low	0.80	0.87	0.83	0.92
	Low	0.72	0.71	0.71	0.85
	Medium	0.67	0.52	0.59	0.71
	High	0.57	0.21	0.31	0.46
	V High	0.87	0.02	0.05	0.24
LR-bal	V Low	0.84	0.83	0.83	0.88
	Low	0.73	0.70	0.71	0.83
	Medium	0.59	0.63	0.61	0.78
	High	0.46	0.60	0.52	0.74
	V High	0.11	0.37	0.18	0.58
DT	V Low	0.83	0.81	0.82	0.89
	Low	0.65	0.75	0.69	0.87
	Medium	0.61	0.46	0.53	0.68
	High	0.54	0.35	0.43	0.52
	V High	0.05	0.01	0.01	0.21

Table 1: k-means to determine Item Price Bands

Method	Pr Band	Prec	Rec	F1	MRR
Baseline	V Low	0.06	0.07	0.06	0.48
	Low	0.11	0.12	0.11	0.41
	Medium	0.16	0.17	0.17	0.40
	High	0.22	0.22	0.22	0.42
	V High	0.46	0.43	0.44	0.58
LR-unbal	V Low	0.59	0.40	0.48	0.54
	Low	0.67	0.32	0.44	0.53
	Medium	0.63	0.40	0.49	0.65
	High	0.63	0.52	0.57	0.74
	V High	0.65	0.89	0.75	0.93
LR-bal	V Low	0.51	0.55	0.53	0.69
	Low	0.60	0.45	0.52	0.64
	Medium	0.58	0.55	0.56	0.73
	High	0.59	0.58	0.59	0.75
	V High	0.76	0.81	0.78	0.87
DT	V Low	0.59	0.32	0.42	0.50
	Low	0.58	0.39	0.47	0.57
	Medium	0.59	0.45	0.51	0.63
	High	0.61	0.47	0.53	0.69
	V High	0.65	0.86	0.74	0.91

Table 2: transac-bal to determine Item Price Bands

Table 3: Evaluation Metrics for Different Price Affinity Models

5 CONCLUSION

In this paper, we discuss a novel approach to incorporate price-related user-item signals into recommender systems to personalize their output. This is done by assigning price bands to items of different types, using historical user-item data to predict user price affinity and using this affinity along with an item price band similarity score to re-rank item recommendations anchored on a (user,

Item Price Band: K-means | User Price Affinity: Balanced Logistic Regression

Legend: V Low (0), Low (1), Med (2), High (3), V High (4)

Item Price Band: Transaction Bal | User Price Affinity: Balanced Logistic Regression

Legend: V Low (0), Low (1), Med (2), High (3), V High (4)

Figure 3: Comparison of Evaluation Metrics for Balanced Logistic Regression with k-means vs transac-bal item price banding

Pearson Correlation	Matrix Factorization
Bed Sheets_2	Bed Sheets_2
Bed Sheets_3	Comforters & Duvets_0
Bed Sheets_1	Bedding Sets_0
Towels & Washcloths_4	Comforters & Duvets_1
Bed Sheets_4	Bath Rugs_2
Bed Pillows_4	Pillowcases_3
Comforters & Duvets_1	Bath Rugs_0
Bedding Sets_2	Bath Rugs_1
Bedding Sets_0	Bed Sheets_3
Bedding Sets_1	Bath Rugs_3

Figure 4: Top 10 similar Product Types, Price Bands to Medium-2 Bed Sheets
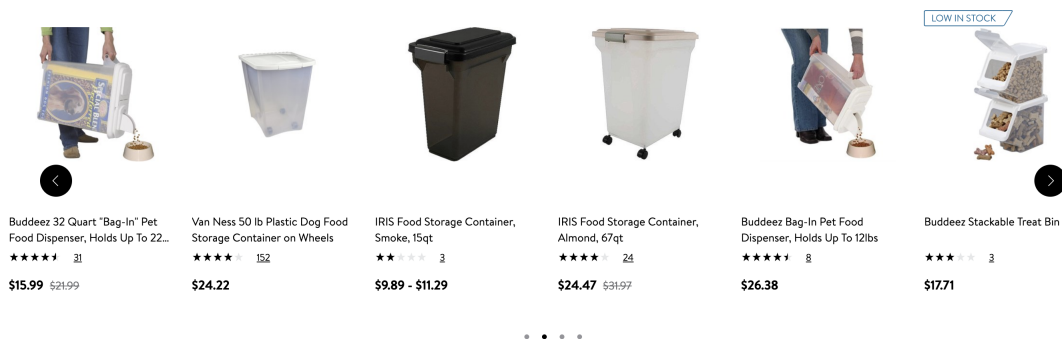
Customers also viewed these products

Figure 5: Viewed also Viewed recommendations for a dog food container item

Item Price Band Method	User Price Affinity Method	NDCG@5	MHR@5	MRR@5	MAP@5
- Relevance -		0.4375	0.8480	0.5418	0.2048
k-means	Baseline	0.4381	0.8493	0.5418	0.2053
k-means	LR (unbalanced)	0.4386	0.8504	0.5417	0.2057
k-means	LR (balanced)	0.4388	0.8508	0.5417	0.2057
k-means	Decision Trees	0.4385	0.8503	0.5416	0.2056
transac-bal	Baseline	0.4384	0.8503	0.5415	0.2054
transac-bal	LR (unbalanced)	0.4388	0.8519	0.5410	0.2057
transac-bal	**LR (balanced)**	**0.4389**	**0.8524**	0.5408	**0.2057**
transac-bal	Decision Trees	0.4386	0.8513	0.5411	0.2056

Table 4: Evaluation Metrics for Different Re-ranking Experiments (using Price Affinity only)

Price Similarity Method	NDCG@5	MHR@5	MRR@5	MAP@5
- Relevance -	0.4375	0.8480	0.5418	0.2048
Pearson Correlation	0.4402	0.8557	0.5403	0.2067
Matrix Factorization	**0.4403**	**0.8565**	0.5400	**0.2067**

Table 5: Evaluation Metrics for Different Re-ranking Experiments (using Price Affinity (best) with Price Similarity)

item) pair. We demonstrate statistically significant improvement in offline ranking metrics after explicitly including price inputs (user price affinity using balanced logistic regression with transaction-balanced price bands; item price similarity using matrix factorization). To compute price affinities, other user-website interaction data such as user's historical search queries can also be used. In the future, we plan to learn embeddings which implicitly encode item price information and user representations such that similarity between user and item embeddings is indicative of price affinity. We can also experiment with other pairwise or listwise learning to rank methods to improve the current pointwise ranking function.

REFERENCES

[1] Grigor Aslanyan, Aritra Mandal, Prathyusha Senthil Kumar, Amit Jaiswal, and Manojkumar Rangasamy Kannadasan. 2020. Personalized Ranking in eCommerce Search. In *Companion Proceedings of the Web Conference 2020*. 96–97.

[2] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson Correlation Coefficient. In *Noise Reduction in Speech Processing*. Springer, 1–4.

[3] Stephen Bonner and Flavian Vasile. 2018. Causal Embeddings for Recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 104–112.

[4] Pei-Yu Chen and Shin-yi Wu. 2007. Does Collaborative Filtering Technology Impact Sales? Empirical Evidence from Amazon.com. *Empirical Evidence from Amazon.Com (July 8, 2007)* (2007).

[5] Pei-Yu Chen, Shin-yi Wu, and Jungsun Yoon. 2004. The Impact of Online Recommendations and Consumer Feedback on Sales. *ICIS 2004 Proceedings* (2004), 58.

[6] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. 7–10.

[7] W Bruce Croft, Donald Metzler, and Trevor Strohman. 2010. *Search Engines: Information Retrieval in Practice*. Vol. 520. Addison-Wesley Reading.

[8] Alexandru M Degeratu, Arvind Rangaswamy, and Jianan Wu. 2000. Consumer Choice Behavior in Online and Traditional Supermarkets: The Effects of Brand Name, Price, and other Search Attributes. *International Journal of Research in Marketing* 17, 1 (2000), 55–78.

[9] Tülin Erdem, Joffre Swait, and Jordan Louviere. 2002. The Impact of Brand Credibility on Consumer Price Sensitivity. *International Journal of Research in Marketing* 19, 1 (2002), 1–19.

[10] Carlos A Gomez-Uribe and Neil Hunt. 2015. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Transactions on Management Information Systems (TMIS)* 6, 4 (2015), 1–19.

[11] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. 2015. E-commerce in your Inbox: Product Recommendations at Scale. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1809–1818.

[12] Asnat Greenstein-Messica and Lior Rokach. 2018. Personal Price Aware Multi-Seller Recommender System: Evidence from eBay. *Knowledge-Based Systems* 150 (2018), 14–26.

[13] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. *arXiv preprint arXiv:1703.04247* (2017).

[14] Sangman Han, Sunil Gupta, and Donald R Lehmann. 2001. Consumer Price Sensitivity and Price Thresholds. *Journal of Retailing* 77, 4 (2001), 435–456.

[15] 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*. 173–182.

[16] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based Recommendations with Recurrent Neural Networks. *arXiv preprint arXiv:1511.06939* (2015).

[17] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 263–272.

[18] Dietmar Jannach, Malte Ludewig, and Lukas Lerche. 2017. Session-Based Item Recommendation in e-Commerce: On Short-Term Intents, Reminders, Trends and Discounts. 27, 3–5 (Dec. 2017), 351–392. https://doi.org/10.1007/s11257-017-9194-1

[19] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated Gain-based Evaluation of IR Techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[20] Anil Kaul and Dick R Wittink. 1995. Empirical Generalizations about the Impact of Advertising on Price Sensitivity and Price. *Marketing Science* 14, 3_supplement (1995), G151–G160.

[21] Gary King and Langche Zeng. 2001. Logistic Regression in Rare Events Data. *Political Analysis* 9, 2 (2001), 137–163.

[22] Thomas N Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint arXiv:1609.02907* (2016).

[23] David G Kleinbaum, K Dietz, M Gail, Mitchel Klein, and Mitchell Klein. 2002. *Logistic Regression*. Springer.

[24] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.

[25] Wei-Yin Loh. 2014. Fifty Years of Classification and Regression Trees. *International Statistical Review* 82, 3 (2014), 329–348.

[26] James MacQueen et al. 1967. Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Vol. 1. Oakland, CA, USA, 281–297.

[27] Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

[28] Barak A Pearlmutter. 1995. Gradient Calculations for Dynamic Recurrent Neural Networks: A Survey. *IEEE Transactions on Neural networks* 6, 5 (1995), 1212–1228.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[30] Steffen Rendle. 2010. Factorization Machines. In *2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.

[31] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian Personalized Ranking from Implicit Feedback. *arXiv preprint arXiv:1205.2618* (2012).

[32] Douglas A Reynolds. 2009. Gaussian Mixture Models. *Encyclopedia of Biometrics* 741 (2009).

[33] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Proceedings of the 10th International Conference on World Wide Web*. 285–295.

[34] Venkatesh Shankar, Arvind Rangaswamy, and Michael Pusateri. 1999. The Online Medium and Customer Price Sensitivity. *Working Paper* (1999).

[35] Brent Smith and Greg Linden. 2017. Two Decades of Recommender Systems at Amazon.com. *IEEE Internet Computing* 21, 3 (2017), 12–18.

[36] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. 2005. Personalizing Search via Automated Analysis of Interests and Activities. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Salvador, Brazil) *(SIGIR '05)*. Association for Computing Machinery, New York, NY, USA, 449–456. https://doi.org/10.1145/1076034.1076111

[37] Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-prod2vec: Product Embeddings using side-information for Recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 225–232.

[38] Kirk L Wakefield and J Jeffrey Inman. 2003. Situational Price Sensitivity: the Role of Consumption Occasion, Social Context and Income. *Journal of Retailing* 79, 4 (2003), 199–212.

[39] Matei Zaharia, Reynold S. Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J. Franklin, Ali Ghodsi, Joseph Gonzalez, Scott Shenker, and Ion Stoica. 2016. Apache Spark: A Unified Engine for Big Data Processing. *Commun. ACM* 59, 11 (Oct. 2016), 56–65. https://doi.org/10.1145/2934664

[40] Yu Zheng, Chen Gao, Xiangnan He, Yong Li, and Depeng Jin. 2020. Price-Aware Recommendation with Graph Convolutional Networks. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 133–144.