Personalized Product Recommendation using Customer Expertise

Stylianos Rousoglou
MS Computer Science
steliosr@stanford.edu

Victoria Toli
Stanford '20
vtoli@stanford.edu

1. Abstract

In this paper, we develop and experiment with a new approach to product recommendation, in which customers' authority relative to specific product categories is quantified and used in recommendation. Given a particular product category, we define "category experts" to be customers who have purchased relevant products, left feedback on those products, and whose comments received positive feedback by others. This way, we are able to discriminate between customers "seasoned" in particular categories and enhance the process of recommending a ranked list of products within a given category to a given user. The approach does not require any external information about users (such as demographics) or specialized product features other than product category labels; it utilizes co-purchasing information and review metadata.

2. Introduction

Recommender systems play an important role in our daily interactions with computers and other electronic devices. They are certainly an important tool for businesses who use them daily to display personalized ads, propose new connections on social media, or suggest new products. The two common approaches to building a recommender system are user-based and item-based recommendations, that consider similar users and items respectively (in terms of recorded preferences) in predicting what a user might also enjoy. However, user-based techniques often rely on external factors (such as personal and demographic information) that may not be available depending on the context, or consider all customers equally important.

Our goal is to build a recommender system that given a particular customer and product category, would return a list of products the customer is likely to purchase and enjoy. To make our approach context-agnostic, we only rely on simple data that is collected by most all modern online retailers and search services (e.g. Amazon, Yelp, etc.), such as co-purchasing information, customer reviews, and community feedback on those reviews (to reward honest reviewers and deter spammers.)

We begin by developing a simple baseline customer similarity algorithm that utilizes co-purchasing data. We manipulate the co-purchasing network to develop a Customer-Category, and subsequently a Customer Network that encodes similarity in tastes between users (both in terms of liking and disliking similar products). We then develop and tune an "expertise" evaluation metric to identify the best-scoring customers given a particular customer and product category and use these "expert" customers to make product recommendations. We use a simple product-similarity evaluation metric computed on the co-purchasing network to measure how likely the customer is to enjoy the recommended products of each approach, present our findings, and discuss possible enhancements and further work that could render our approach more useful.

3. Related Work & Critique

Traditional approaches to data filtering

While the development of the internet led to a massive increase in online information, it also enhanced the difficulty of navigating and searching for information, from a user's perspective. Filtering algorithms, on a content or collaborative level, constitute one way to deal with this problem. The former pertains to recommending objects based on their descriptions. Nevertheless, description storing is an overwhelming task for an exponentially growing number of objects. Simultaneously, for users who have selected an insufficient number of object types, this content-based approach hinders performance.

Alternatively, collaborative filtering consists of two categories: one that examines user-to-user similarity and one that focuses on item-to-item resemblance. Item-based filtering supposes that users tend to choose items similar to ones they have *already selected*, while user-based filtering assumes that *similar users* will tend to select *similar items*.

Although traditional collaborative approaches cover the obvious shortcomings of content-based methods, they have significant drawbacks, pertaining to sparsity, scalability, and the gray sheep problem (Ha et al, 2016; Lee et al., 2004; Shi et al., 2014; Su et al, 2009). The issue of sparsity describes the situation when user-to-user or item-to-item relations are biased: rarely used items are not reflected in the recommendations. Scalability is problematic when a recommendation system needs to compute similarity and preference for an enormous set of objects or users. Lastly, the gray sheep problem refers to the situation when a user's preferences are atypical or inconsistent with others'. Those three considerations hamper both performance and ease of implementation.

Recent developments in collaborative filtering

Newer approaches to collaborative recommendation systems have accounted for the aforementioned limitations. Ha and Lee (2016) proposed an item-network-based collaborative filtering, which constructs an item network based on users' item selection history and computes three types of node centrality: betweenness, closeness, and degree. Nodes' betweenness is used to identify significant objects in a user's item graph. Afterwards, closeness and degree centrality of those significant nodes give each item a preference score, from each user's perspective. A recommendation relies on sorting those preference scores.

An alternative approach utilized user comments in networks where those are available (Wang et al, 2010). The premise is that comments convey highly reliable information about users' behavior. Although this has largely been leveraged in a subset of networks (namely, social media), the method makes an interesting point: it does not assume that all comments are of equal importance; it computes an authority index for each user, based on the number of times she has been quoted or replied to, by employing a variant of the PageRank algorithm (Brin et al, 1998). Yet, is even more relevant and useful is a *genera*l index of user authority, irrespective of whether user comments are available in a network. Such index will be proposed in the approach section.

4. Dataset

Data format

We are using the *Amazon product co-purchasing network metadata* dataset,[1] which consists of 548,552 products, and 7,781,990 user reviews of purchased products, as well as binary feedback (helpful/not helpful) on those reviews provided by the Amazon community. Each product is given a unique product ID and assigned a category as well as several subcategory labels. Users are also given unique IDs that are associated with each review that has been posted.

Data extraction

To be able to work with the data set, significant preprocessing had to be performed. Our data extraction algorithm returns two structures, one pertaining to customers and one to products. For the customers data structure, we are mapping customers to products purchased, reviews, and review feedback:

***key**: Cust Id, **value**: {Prod Id: [rating, votes, helpful]} of products purchased*

The second data structure that we construct is a product map storing all product information available, including the title, categories (option differ depending on the group, e.g. Mystery & Thrillers, Science Fiction, etc. for books), and which customers have purchased and written a review of each product:

***key**: Prod Id, **value**: {title, group, categories (set of category Ids), reviews (list of customer Ids)}*

However, some products are listed under multiple categories, each of which associated with multiple subcategories. To simplify such complicated labelling schemes while maintaining a certain degree of specificity in the description of each product, we only maintained the most general genre-specific category labels. Labels referring to specific actors, locations, or periods, were purged for simplicity.

Data reduction

When we started experimenting with graph folding of the multimodal networks we created, it became evident that our machines lacked the computational power to run algorithm on networks constructed using the entirety of our dataset. Specifically, by tracking the runtime performance, we projected that folding the *Customer-Category Network* into the *Customer Network* would have lasted ~ 150 hours and required ~ 70 GB of memory on our 2016 Macbook Pro machines. To reduce the dataset and run our algorithms successfully, we decided to produce a *random sample* of our customers that accounts for ~ *2%* of the initial 550,000 or so customers. To maximize our chances of obtaining meaningful results, we only sampled customer who had purchased at least 3, and at most 100, products.

---

[1] https://snap.stanford.edu/data/amazon-meta.html

5. Approach

5.1 Network Construction

Our ultimate goal is to create a hybrid recommendation approach, which combines user similarity with a model of ascribing higher importance to certain customers given a particular product category. To do so, we create a simple Customer - Product network; we transform it into a Customer - Category network; and we finally fold it into a unimodal Customer network.

Customer-Product Network

The bimodal Customer-Product network we constructed consists of customer and product nodes. Each customer is connected to each product they have purchased. Nevertheless, we believe that liking or disliking a product purchased are both important in understanding customer tastes and predicting preferences. Since snap does not facilitate edge weight storage, we stored the ratings as values in a customer-product weight map. As such, the range of weights is $w \in \{1, 2, 3, 4, 5\}$.

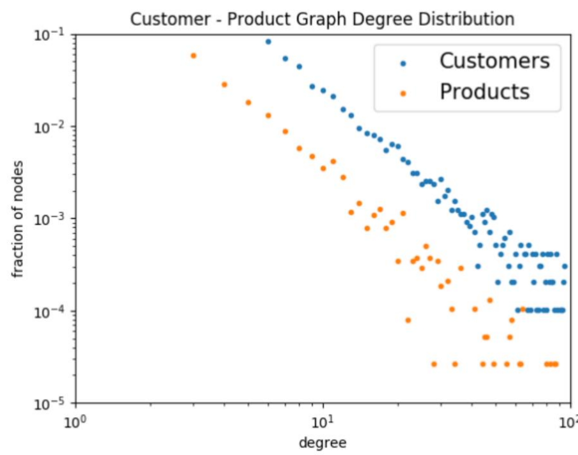

Figure 1: C-P Network Unweighted Degree Distribution

Customer-Category Network

To transform the Customer-Product to a bimodal Customer-Category network, we contract all product nodes of a specific category into a single super node; the weight of the edge between given customer and a category super node is equal to the average weight of edges between the customer and purchased products of said category. If a product belongs in multiple categories, e.g. a book is labelled both *Mystery & Thrillers* and *Science Fiction*, an edge is added between each customer who purchased it and both (*book - Mystery & Thrillers)* and (*book, Science Fiction)* category nodes. We once again maintain weights in a map; *the weight between customers* i and *category* j are computed using the formula:

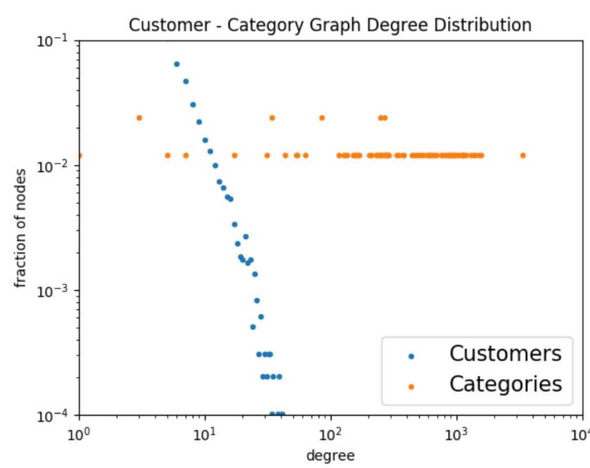

Figure 2: C-C Network Unweighted Degree Distribution

$$w_{i,j} = \frac{1}{5 \times |N_{CP}(i, p \mid p \in j)|} \sum_{\substack{p \in N_{CP}(i) \\ and\ p \in j}} a_{ip}$$

i.e. the *normalized average weight* between *customer* i and products that belong to *product category* j in the *original Customer-Product graph.* As such, the range of weights is $w \in [0, 1]$.

Customer Network

Finally, the Customer Network is generated as a unimodal projection of the Customer - Category Network. Specifically, user nodes in the CC Net become nodes in the C Net. Edges are then constructed as follows: for each pair of customers, we find all common neighbors in the CC Net (which correspond to product categories.) For each of those categories, if the weights between the users and the category node in CC Net are both above, or both below, a liking and a disliking threshold, respectively, we consider the customers' tastes in the particular category similar and add $+1$ weight between the customer nodes in C Net. After repeating the process for all customer pairs, we normalize the weights to rescale them in the range $w[0, 1]$. The weight between two customers i and j in the C Net is computer as follows:

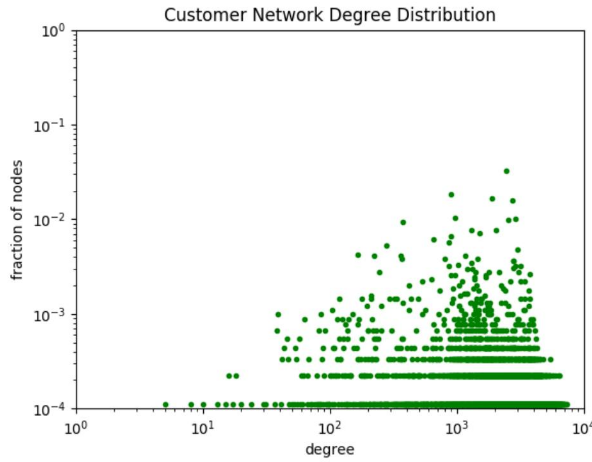

Figure 3: C Network Unweighted Degree Distribution

$$w_{i,j} = \frac{1}{|\, N_{CC}(i) \cap N_{CC}(j)|} \sum_{cat \in N_{CC}(i) \cap N_{CC}(j)} I(cat, i, j)$$

i.e. where $I(c, i, j)$ an indicator function such that:

$$I(cat, i, j) = \begin{cases} 1 \; if \; w_{CC}(i, cat) \geq 0.7 \; and \; w_{CC}(j, cat) \geq 0.7 \\ 1 \; if \; w_{CC}(i, cat) \leq 0.3 \; and \; w_{CC}(j, cat) \leq 0.3 \\ 0 \; otherwise \end{cases}$$

As such, the range of weights is $w \in [0, 1]$. We proceeded to purging edges with weights less that 0.5, as they would make similarity computation unnecessarily more time-consuming. The runtime of constructing the Customer Network is $O(|Cust|^2|Cat|)$, which is the main obstacle we encountered when attempting to construct a Customer Network using the entire dataset.

Summary of Networks

	CP Net	CC Net	C Net
Nodes	47,021	9,141	9,057
Edges	75,407	44,040	**8,995,947**
Customers	9,057	9,057	9,057
Products	37,964	-	-
Categories	-	84	**-**
Av. Clust. Coeff.	-	-	0.76

5.2 Approaches to Ranking Customers

Baseline User Similarity

To compute user similarity, we implemented a slightly modified version of the Jaccard measure of similarity. For customers i and j, and neighbor sets $N(i)$ and $N(j)$, the similarity is computed by the following formula:

$$J(i,j) = \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|} = \frac{|N(i) \cap N(j)|}{|N(i)| + |N(j)| - |N(i) \cap N(j)|}$$

In our method, for a given pair of users i and j, the Jaccard Index was computed twice on the Customer-Product Graph: first, $N_{liked}(i), N_{liked}(j)$ contain the products users i and j actually *liked*, defined by a user rating *above* a predefined liking-threshold. This way, similarity is not merely defined on the basis of jointly purchased products, but jointly enjoyed products, which we find to be a more representative indicator of similarity. In the second case, $N_{disliked}(i)$, $N_{disliked}(j)$ contain the products user i and j *disliked*, defined by a user rating *below* a predefined disliking-threshold. Despite the fact that we believe there is significance in users commonly disliking a product, we decide to give commonly enjoyed products a higher importance. Thus, the final Jaccard user similarity was computed as follows:

$$J(i,j) = 2 \times J_{liked}(i,j) + J_{disliked}(i,j)$$

To recommend products using this similarity definition, we first identify (10) customers j with the highest $J(i,j)$. From those customers' purchased items, we recommend products of the requested category that have received the highest ratings by the same customers. Hence, this standard approach offers (10) product suggestions, leveraging user similarity and product ratings.

Customer Expertise

Our contribution to the Product Recommendation literature is a more expressive similarity index between users which, while accounting for multiple factors, does not require any user information other than that available in a Co-Purchasing dataset. After developing such an index, we are interested in exploring how Customer Experts can be found given a query from customer i for products in category k, and used to make a personalized recommendation to customer i.

The factors accounted for were decided empirically and added incrementally to the Customer Expertise, in order to account for a multitude of parameters computable using the Co-Purchasing Net alone (See 6.1 for results). First, weights between customer i and neighboring customers in the C Net are a measure of common interests between the users, which is relevant in looking at other customers to recommend products from. The "experience" of each customer j in product category k, as measured by the fraction of products customer j purchased in category k relative to all the products they purchased, is also relevant, since users who have purchased many products in category k can generally rate and compare such products more effectively. To discriminate against fake reviews and spam, the "helpfulness" of customer's reviews, as measured by the fraction of "helpful" votes to total feedback votes, also had to be accounted for. Finally, to discriminate between experienced and new users, the total number of purchases and reviews is accounted for as a fraction of a customer's purchases relative to all purchases.

In summary, the factors settled upon are:

1. The commonality of interests between customer i (querying customer) and all other customers, as measured by the weights between users in the C Net (see Section 5.1)
2. The fraction of products purchased by each customer j in product category k
3. The "helpfulness" of customers' reviews as measured by feedback from other users
4. The relative total number of products each customer j has purchased and reviewed.

For each customer j, the "expert" score given querying customer i and product category k is computed as follows:

$$E_{i,k}(j) = w_{CNet}(i,j) \times \frac{N_{CP}(j) \cap \{all\ p \in k\}}{N_{CP}(j)} \times \left(\frac{1}{|N_{CP}(j)|} \sum_{p \in N_{CP}(j)} \frac{helpful(j,p)}{total(j,p)}\right) \times \ln(|N_{CP}(j)|)$$

where w_{CNet} the edge weight in the C Net, $helpful(j,p)$ the number of "helpful" feedback votes on customer j's review of product p, $total(j,p)$ the total feedback votes on that same review, and E_{CP} the edges in the CP Net.

To recommend products using this similarity definition, we first identify (10) experts j with the highest $E_{i,k}(j)$. From those customers' purchased items, we recommend products of the requested category that have received the highest ratings by the experts.s

6. Experiments & Results

6.1 Recommending Experts

Table 1 represents a sample run of the expertise computation algorithm for $customer\ Id = 118$ and product category "Comedy DVD", presenting some empirical evidence for the necessity of incorporating multiple factos in computing expertise. The IDs of the experts are shown, along with the expertise score and the values of the various factors taken into account by the final expertise model.

Column 1 presents the experts returned when only factor 1 (commonality of interests) is considered. Although customer 18 seems to have liked (and disliked) categories similarly (as all experts on Table 1 have, as reflected by their common the 1.0 score in factor 1), it's worth noticing that customer 18, who is returned as the *highest-scoring* expert in Column 1, *has only made* 3 *purchases in total.* In contrast, Expert #2 has 44 recorded purchases.

Column 2 presents experts returned when factors 1, 2, and 4 are considered in the computation. Customer 18 is now ranked #4, while all other experts are different, and have purchased more products in total. However, we notice that for Expert #1, customer 2299, only 0.28 of community feedback consider their reviews "helpful". Such significantly low percentage signifies that customer 2299 may be a spammer, a paid reviewer, or simply a low information content reviewer; using him to recommend products is not desirable. Column 3 present the results of the ultimate expertise model. Although the experts returned have *not* purchased as many products in the category of interest relative to their total purchases as experts in Column 2 have, they have a “helpfulness” fraction of at least 0.8; also, they have all purchased *at least* 6 products, a standard that 2 out of 5 experts in Column 1 fail to meet.

“Expertise” factors	**1**	**1,2,4**	**1,2,3,4**
Expert 1:	18	2299	4405
Score:	7.21	9.78	9.36
Info:	[1.0, 0.66, 0.4, 3]	[1.0, 0.55, 0.28, 11]	[1.0, 0.67, 1.0, 6]
Expert 2:	23	171	3934
Score:	7.21	9.43	8.57
Info:	[1.0, .023, 1.0, 44]	[1.0, 0.71, 0.87, 7]	[1.0, 0.57, 1.0, 7]
Expert 3:	44	6817	1474
Score:	7.21	9.36	8.50
Info:	[1.0, 0.17, 0.67, 6]	[1.0, 0.67, 0.3, 6]	[1.0, 0.48, 0.8, 21]
Expert 4:	86	18	171
Score:	7.21	6.67	8.17
Info:	[1.0, 0.15, 1.0, 13]	[1.0, 0.67, 0.4, 3]	[1.0, 0.71, 0.87, 7]
Expert 5:	106	101	5508
Score:	7.21	4.56	8.15
Info:	[1.0, 0.25, 1.0, 4]	[1.0, 0.27, 0.86, 11]	[1.0, 0.45, 1.0, 11]

Table 1: Top 5 Experts for user Id 118, product category 'DVD' + 'Comedy', using factor 1, factors 1,2,4, and factors 1,2,3,4, respectively (see 5.2)

6.2 Recommending Products

To quantitatively evaluate the products returned against a simple baseline, we use a product-similarity metric derived from the Co-Purchasing network (CP Net). Specifically, we used cosine similarity to quantify the degree to which 2 products were equally liked (or disliked) by customers who purchased them both. As such, this similarity metric doesn’t measure intrinsic similarity of the products, but customer rating correlation between the two. Formally,

$$S(p_1, p_2) = \frac{\sum_{i \in N_{CP}(p_1) \cap N_{CP}(p_2)} r_i^{(1)} \times r_i^{(2)}}{\sqrt{\sum_{i \in N_{CP}(p_1) \cap N_{CP}(p_2)} {r_i^{(1)}}^2} \times \sqrt{\sum_{i \in N_{CP}(p_1) \cap N_{CP}(p_2)} {r_i^{(2)}}^2}}$$

where $r_i^{(1)}$the rating assigned by customer i to product 1. It’s important to note that **the above similarity matric is evaluated on the *entirety* of the co-purchasing network** (100% of the data) instead of the sample used in building networks, to allow for maximal expressivity of the metric. Now consider two lists of products, π_1 and π_2, We define average similarity of those two lists as:

$$S_{AV}(\pi_1, \pi_2) = \frac{1}{|\pi_1| \times |\pi_2|} \sum_{i \in \pi_1} \sum_{j \in \pi_2} S(i,j)$$

Let us now examine an example of what product recommendations the customer similarity and customer expertise-centered approaches might generate. The following table presents the results returned by each approach, along with a list actual customer purchases, by customer 118.

ID: 118	**Purchased products** (π_p)	**Baseline recommendations** (π_b)	**Expertise recommendations** (π_e)
	The Big Chill	*Anywhere But Here*	*Santa Claus the Movie (Full Screen Edition)*
	The Hollywood Knights	*Vampire's Kiss*	*But I'm a Cheerleader*
	Somewhere in Time	*Haiku Tunnel*	*Harold and Maude*
	Mystic Pizza	*The Mexican*	*Almost Famous*
	The Big Chill (15th Anniversary Edition)	*Holy Smoke!*	*Bruce Campbell vs. Army Of Darkness - The Director's Cut (Official Bootleg Edition)*
	The Great Outdoors	*Say It Isn't So*	*Almost Famous Untitled - The Bootleg Cut (Director's Edition*
	Rivethead: Tales from the Assembly Line	*An Everlasting Piece*	*Army of Darkness*
	The Wild One	*The World According to Garp*	*Army of Darkness (Boomstick Edition)*
	Somewhere in Time (Collector's Edition)	*Jackass - The Movie (Full Screen Special Edition)*	*Galaxy Quest - DTS*
	-	$S_{AV}(\pi_p, \pi_b) = \mathbf{0.459}$	$S_{AV}(\pi_p, \pi_e) = \mathbf{0.644}$

Table 2: Recommendations of baseline algorithm and experts algorithm for customer 118, category "Comedy DVD"

In order to explore how much better the expert recommendations might perform against our baseline customer similarity recommendations, we ran both algorithms on a sample of 2,000 random customer – category combinations, obtained $S_{AV}(\pi_p, \pi_b)$, $S_{AV}(\pi_p, \pi_e)$ results, and evaluated the average improvement in similarity obtained by using the experts, as well as the fraction of sample runs that experts recommended more correlated products overall. Results are summarized in Table 3.

	Baseline Recs.	**Expert Recs.**
$\boldsymbol{S_{AV}}$	0.572	0.744
$\boldsymbol{S_{AV}(\pi_p, \pi_e) > S_{AV}(\pi_p, \pi_b)}$	0.34	0.66
Av. Similarity Gain	**+ 17.2%**	

Table 3: Summary of results on 2,000 sample customer / category combinations, and net similarity gain results. Each run took ~ 10-15 seconds, for a total of ~ 7 hours of total runntime

7. Discussion & Limitations

Running both algorithms on 2,000 random customer / category pairs is a sufficient sample to provide a reasonable overview of the difference in performance between the two approaches outlines in Section 5. As presented in Table 3, expert recommendations dominated simple customer similarity recommendations by an average similarity of $\mathbf{+0.172}$. In $\sim\mathbf{66\%}$ of all sample runs, the expert recommendations performed better. We proceed to discuss the implications of these results and suggest ways that the model could be enhanced and expanded.

First, we were interested in exploring how the customers querying our algorithm for product recommendations may themselves affect the results. We noticed that for some customers, expert recommendations were performing worse than baseline recommendations. After manually inspecting such examples, we performed sample runs on 1,000 customer / category pairs of two kinds: customer / category pairs for which customers had already purchased at least 1 product of the given category; and customer / category pairs for which customers hadn't. The results presented in Table 4 indicate a significant disparity between samples of the first and second kind. Specifically, the expert recommendation worked **~ 22% better** in cases where customers had already purchased one or multiple products of given category.

An indicative example of the problem is shown in Table 5 (See Appendix A), where the list of the user's purchased products is highly redundant, and repeated information confers little improvement to our understanding of the user (i.e. the list includes products that are almost identical, such as *Pride And Prejudice (Scholastic Classics)* and *Pride and Prejudice (Penguin Classics)*). Hence, for both recommendations, the results have a relatively low similarity to the purchased items. At the same time, if all products purchased by the customer belong to a single category, which is inherently vastly different from the requested category, both recommendations are bound to perform poorly.

	Average similarity to purchased products
Condition 1: user has *no*t purchased any product from the requested group-category.	0.589
Condition 2: user has purchased at *least one* product from the requested group-category.	**0.813**

Table 4: Effect of customer purchases on the quality of user expertise-centered recommendation

There are several limitations in our approach worth addressing. The main limitation we encountered throughout all phases of our project was a lack of enough computational resources. The fact that we needed to reduce our data by ~ 98% (See Section 4) meant limiting the expressiveness of our model and results; significantly less customer – product data imply less accurate estimates of customer similarity and customer expertise. Using multiple cores and parallelizing computation could have sped-up network operations and allowed us to use more data.

For a more rigorous evaluation of the extent of success of using product category experts to recommend products, we could use an active data collection scheme that would collect feedback from product recommendations to customers and which could then be used to refine the model. In simulating such a scheme in this project, one option would have been to use temporal information of customer purchases to split purchases between training and testing sets; however, that approach would have limited our training data even more, so we decided to not follow it.

Utilizing user feedback on information content is already widely popular in contexts such as social media (where the relative amount of mainly positive feedback determines the visibility and prominence of user-generated content such as Facebook comments or Tweets.) In developing the customer expertise metric, we wanted to capture the significance of such feedback available in the Amazon product metadata dataset. Although it's clear that utilizing expertise score to recommend products performs well in a significant fraction of queries, a more expressive metric than the binary "helpful" vs. "not helpful" would likely cater to the development of a more fine-tuned model.

8. Contributions

Stelios: Abstract + Intro, Customer-Product Net, Customer-Category Net, Customer Net, Expertise (Metric, Experiments & Recommendation), Limitations

Victoria: Abstract + Intro, Related Work, Baseline Customer Similarity (Metric, Experiments & Recommendation), Product Similarity Metric + Experiments, Limitations

9. Github

All code, helper files and graphs used, are available at github.com/steliosrousoglou/224W.

10. References

Ha, T., & Lee, S. (2017). Item-network-based collaborative filtering: A personalized recommendation method based on a user's item network. Information Processing & Management, 53(5), 1171–1184. https://doi.org/10.1016/j.ipm.2017.05.003

Jia Wang, Qing Li, Yuanzhu Peter Chen, and Zhangxi Lin. 2010. Recommendation in Internet forums and blogsZ. Association for Computational Linguistics, Stroudsburg, PA, USA, 257-265.

Koohi, H., & Kiani, K. (2017). A new method to find neighbor users that improves the performance of Collaborative Filtering. Expert Systems with Applications, 83, 30–39. https://doi.org/10.1016/j.eswa.2017.04.027

Ma, X., Li, B., & An, Q. (2013). A Network-Based Approach for Collaborative Filtering Recommendation. In Behavior and Social Computing (pp. 119–128). Springer International Publishing. https://doi.org/10.1007/978-3-319-04048-6_11

Su, X., & Khoshgoftaar, T. M. (2009). A Survey of Collaborative Filtering Techniques. Advances in Artificial Intelligence, 2009, 1–19. https://doi.org/10.1155/2009/421425

U, L., Chai, Y., & Chen, J. (2018). Improved personalized recommendation based on user attributes clustering and score matrix filling. Computer Standards & Interfaces, 57, 59–67. https://doi.org/10.1016/j.csi.2017.11.005

Appendix A

User ID: 49	**Purchased products**	**Purchased products**	**Baseline recommendations**	**Expertise Recommendations**
	Pride and Prejudice	Pride And Prejudice (Everyman's Library)	Betrocks Guide to Landscape Palms	Barbie Doll Fashion: 1968-1974 (Barbie Doll Fashion)
	Pride And Prejudice (Scholastic Classics)	Pride and Prejudice (Dover Thrift Editions)	Betty Crocker's Cookbook: Bridal Edition	Barbie Fashion, 1959-1967 (Barbie Doll Fashion)
	Pride and Prejudice (Audio Editions)	Pride and Prejudice (Bookcassette(r) Edition)	Victoria: At Home with White: Celebrating the Intimate Home	The New Cottage Home
	Pride and Prejudice (Penguin Classics)	Pride and Prejudice (Modern Library Classics)	The Art of Polymer Clay: Designs and Techniques for Making Jewelry, Pottery and Decorative Artwork	The Camerer Cuss Book of Antique Watches
	Pride and Prejudice, Third Edition (Norton Critical Editions)	Pride and Prejudice (Modern Library)	Scented Geraniums: Knowing, Growing, and Enjoying Scented Pelargoniums	Amazing Gracie: A Dog's Tale (Thorndike Core)
	Pride and Prejudice (Oxford World's Classics)	Pride and Prejudice (Oxford Illustrated Jane Austen)	The Polymer Clay Techniques Book	Furniture Treasury (Furniture Treasury)
	-	-	$S_{AV}(\pi_p, \pi_b) = 0.420$	$S_{AV}(\pi_p, \pi_e) = 0.610$

Table 5: Personalized Recommendation for "Home & Garden" Book