Discovering Taste and Interests on Etsy

Anonymous Author(s)
Affiliation
Address
email

Abstract

We use Latent Dirichlet Allocation (LDA) to infer users' categorical and stylistic interests in an online marketplace. The inferred interests are used to recommend other users with similar interests to follow. We also experiment with hashing methods to compute the user recommendations in the map-reduce framework.

1 Introduction

Etsy[1] is an online marketplace where individuals around the world sell handmade and vintage goods. With over 1 million active sellers and 30 million active listings at any given time, it often becomes overwhelming for users to find relevant items to browse through or purchase. The products on Etsy are characterized not only by categories (i.e., a desk or purse), but also by style (e.g., modern, cottage, geometric, etc.). In general, we find that Etsy users are willing to browse through thousands of categorically relevant products, in order to find one that suits their style.

Once a user's categorical and stylistic interests have been identified, they can be utilized for a wide array of recommendation tasks. However, recommending products based on style can be challenging. While it is common for sellers to label their products with information about the category or materials, very rarely do they indicate the style of the product. Furthermore, the average user may not even know how to describe their style preference in words (but will recognize the style when they see it in a product). Different people may also have conflicting definitions of a particular style, making it difficult to create style labels.

In this paper, we show how a user's stylistic and categorical preference (summarized as a user's "interest profile") can be learned in an unsupervised manner using topic modeling algorithms (section 2). These interest profiles are then used to recommend users with similar taste to over 30 million Etsy users (section 2.1). Finally, we share some experimental results comparing our user recommendation and nearest-neighbor algorithms with other baseline approaches (section 3).

2 User Recommendations at Etsy

Our model is based on Latent Dirichlet Allocation, an unsupervised, probabilistic, generative model for discovering latent semantic topics in large collections of text (see e.g., [1].). Our use of LDA is based on the premise that Etsy users with similar taste will like similar products. On Etsy, a user can *favorite* a product, which is a strong indication that the product appeals to their interests (and a more common event than a purchase). We treat each user as a "document," and each of that user's favorite product ids as "words." The goal of learning is to infer topics from the favorited products that commonly co-occur in users' favorite lists. Each topic is characterized by a distribution over products, which we interpret as a particular "interest."

We estimated the model on those users with at least 50 favorite products. There are approximately a million such users, and the resulting data is small enough that the model could be estimated on one machine. We used a computer with 100Gb of RAM and a sixteen CPU cores to run the multithreaded implementation of collapsed Gibbs sampling (see e.g., [4]) from mallet[2]. The topic model consists

[1] http://www.etsy.com
[2] http://mallet.cs.umass.edu

Figure 1: Different interests: Each topic show products across several different categories, but share a common interest, identified as follows: (A) foxes, (B) tentacles, and (C) The Legend of Zelda.

Figure 2: Different styles within the same category: Both topics show products from the same furniture category, but (A) shows mid-century modern style, while (B) shows a rustic/wooden style.

of 1000 topics and the estimation takes approximately a day. Since the resulting model consists of a subset of all the products on Etsy, any products which are not included are treated as the nearest included product by tf-idf distance of the text attributes of the product (the title and tags).

We present some examples of topics that LDA discovers by showing the products that were given the highest weights in each topic. These topics capture both categories and styles: Figure 1 shows topics that center around a certain kind of interest, while spanning many different categories. Figure 2 shows topics that represent different styles in the same furniture category. Finally, Figure 3 shows topics that represent different styles across different categories.

The LDA model also characterizes each user by a vector of weights that indicate how strongly a user identifies with each topic, or interest. Thus we use these vectors to summarize a user's interest. In the next section, we compare how similar one user is to another by looking at how similar the corresponding topic weights are.

2.1 Generating User Recommendations

We conduct a nearest neighbors search in the topic simplex to find candidates for user recommendations. While we use Euclidean distance throughout, these methods can be applied with other metrics. The problem of nearest neighbor search on the simplex was considered in [3], but focus was on the setting in which all user vectors fit into one machine in memory. In contrast, we consider using map-reduce to compute the nearest neighbors in parallel. so that we may scale to billions of users and high dimensional topic models, without memory or running time becoming an issue.

Figure 3: Different styles across different categories: Both clusters contain products across many categories, but (A) shows a geometric/modern style, while (B) shows a rustic/cottage style.

Model Name	# Negative	# Neutral	# Positive	Total # Ratings	Weighted Average
LDA	196	278	440	914	2.27
Cosine Similarity	361	357	315	1033	1.96
Triadic Closure	480	248	138	866	1.61

Table 1: Comparison of LDA with two popular baseline methods. The weighted average attributes 1 point to negative ratings, 2 points to neutral ratings, and 3 points to positive ratings.

Examining every pair of users to determine the distance (the "brute force" approach) is unfeasible due to the large number of users. Therefore we consider two hashing methods. The idea is to hash each users topic vector, then to compute nearest neighbors within each hash bucket. This gives an approximate nearest neighbors method where the overall time complexity is dictated by the size of the largest hash bucket, which we can manage directly. Likewise the resulting algorithm is a natural candidate for parallelization using map-reduce. The elements of each hash bucket can be grouped together and then the buckets can be processed in parallel.

2.1.1 Locality Sensitive Hashing

Locality Sensitive Hashing (LSH) works by splitting the input space into several cones and numbering each one (see e.g., [2]). Then the elements contained in each cone are mapped to its corresponding number. We generate m random planes which pass through the origin (in d-dimensions the normal vector to each plane is generated from a d-dimensional isotropic Gaussian), denote these normal vectors v^i then map each point $\theta \in \mathbb{R}^d$ to

$$H_{\text{LSH}}(\theta) = \sum_{i=1}^{m} 2^{i-1} \mathbf{1}\left\{\theta^T v^i \geq 0\right\}.$$

Thus each point is hashed to an m-bit integer. In our experiment we use $m = 16$. Finally, note that while this algorithm maps some nearby points to the same integer, there may be points which are close but separated by one of the planes. In order to mitigate this problem, we perform the hashing multiple times, compute nearest neighbors in each hash bucket, then combine the results of the different hashings.

2.1.2 "Top-K hashing"

We propose a second hashing method which takes advantage of the sparsity we anticipate in the topic mixture vectors from LDA. It is plausible that the nearest neighbors to a certain user will share some subset of top-k interests. Therefore we map each topic vector to the set of all pairs of topic indices from the top-k topics. The reason for taking pairs rather than individual topic indices is to make more specific hash buckets which will have smaller size. Note that in this method each vector gets mapped into several hash buckets. We compute nearest neighbors in each bucket, then combine these across buckets and take the nearest neighbors from among those candidates.

3 Experiments

We first compare our topic modeling approach to two other user recommendation heuristics in a user study: 1) *Cosine Similarity*: Represent users as a bag of favorite products, and recommend other users with high cosine similarity, and 2) *Triadic Closure*: Recommend other users who also follow the same users. In our user study, 135 Etsy users were presented a randomly interleaved list of 30 recommended users (10 recommendations from each of the 3 models). Users were asked to rate each recommendation as negative ("I would not want to follow this user's activity"), neutral ("I wouldn't mind following this user"), or positive ("I would like to follow this user"). The results in Table 1 show that the LDA approach was the clear favorite.

We also compare the performance of the above nearest-neighbor search methods on the grounds of their approximation quality and computational expense. For this experiment we used a test set of approximately 800,000 users. Their topic vectors were inferred from a previously trained LDA model, with 1000 topics. In order to establish the performance of the above hashing methods we compare to the exact nearest-neighbors. Since generating these for the entire set of users is computationally expensive, we restrict the experiment to a subset of 300 Etsy employees.

Method	Number of Comparisons	20-NN Precision
Brute force	260000000	1.0
LSH-10	17711	0.37
LSH-21	38498	0.56
LSH-45	79141	0.80
TopK-5	45195	0.58
TopK-7	96630	0.68
TopK-10	197762	0.75

Table 2: Computational expense vs precision of the retrieved 20 nearest neighbors.

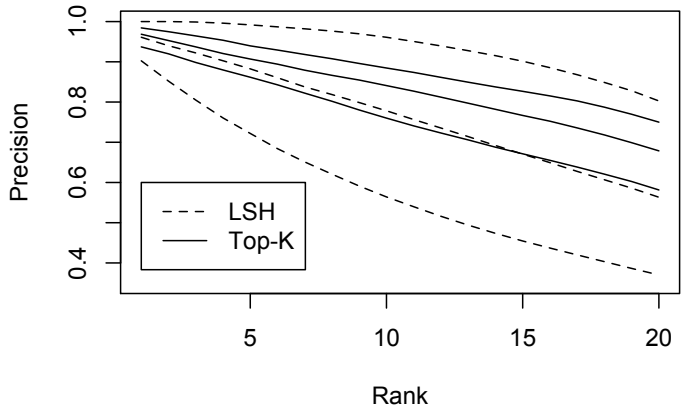

Figure 4: Precision at rank graph for experimental methods. In both cases the higher curves correspond to higher parameter settings (i.e., the top curve is LSH-45).

Both hashing methods are parameterized in a way which allows control over the number of hash bins that each user is assigned to, and we test three settings for each method. For the LSH method we use 16 bit hash keys, and 10, 21 and 45 hashes per user respectively. For the top-k hashing we set k to 5, 7 and 10 and hash the vectors according to pairs of topics in the top k (leading to 10, 21 and 45 hashes per vector). We report the number of pairwise comparisons between user vectors that are computed in Table 2, and then the precision at rank n, for the 20 nearest neighbors in Figure 4. The results demonstrate that for our LDA model, both hashing methods perform adequately, although the LHS method seems to perform better than the top-k hashing method, both in terms of computational cost and the quality of the approximation.

4 Conclusion

We described an LDA model for finding users with similar taste on Etsy. We are currently experimenting with the resulting recommendations, and intend to incorporate product content features, as well as prior knowledge on the user (e.g., gender, shopping intent) in future models.

References

[1] D. M. Blei, A. Ng, and M Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[2] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *SCG*, pages 253–262. ACM, 2004.

[3] K. Krstovski, D.A. Smith, H. M. Wallach, and A. McGregor. Efficient nearest-neighbor search in the probability simplex. In *ICTIR*, pages 22:101–22:108, 2013.

[4] I. Porteous, A. Asuncion, D. Newman, P. Smyth, A. Ihler, and M. Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *ACM SIGKDD*, pages 569–577, 2008.