A Model-based Embedding Technique for Segmenting Customers

Srikanth Jagabathula
Stern School of Business, New York University, New York, NY 10012,sjagabat@stern.nyu.edu

Lakshminarayanan Subramanian, Ashwin Venkataraman
Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, {lakshmi,ashwin}@cs.nyu.edu

We consider the problem of segmenting a large population of customers into non-overlapping groups with similar preferences, using diverse preference observations such as purchases, ratings, clicks, etc. over subsets of items. We focus on the setting where the universe of items is large (ranging from thousands to millions) and unstructured (lacking well-defined attributes) and each customer provides observations for only a few items. These data characteristics limit the applicability of existing techniques in marketing and machine learning. To overcome these limitations, we propose a *model-based embedding technique* which takes the customer observations and a probabilistic model class generating the observations as inputs, and outputs an *embedding*—a low-dimensional representation in Euclidean space—for each customer. We then cluster the embeddings to obtain the segments. Theoretically, we derive precise necessary and sufficient conditions that guarantee asymptotic recovery of the true segments. Empirically, we demonstrate the speed and performance of our method in two real-world case studies: (a) upto 84% improvement in accuracy of new movie recommendations on the MovieLens dataset and (b) upto 8% improvement in the performance of similar product recommendations algorithm on an offline dataset at eBay. We show that our method outperforms standard latent class, empirical bayesian and demographic-based techniques.

Key words: Segmentation, Estimation/statistical techniques, embedding, missing data

1. Introduction

'Customer segmentation' is the practice of grouping customers into non-overlapping segments (or clusters[1]) such that customers in the same segment have similar needs and preferences. It is now a ubiquitous practice carried out by firms. It allows them to effectively customize their product offerings, promotions, and recommendations to the particular preferences of each segment (Smith 1956). Segmentation also subsumes *personalization* as a special case by placing each customer into a separate segment of her own. Personalization has gained a lot of traction recently. Yet, in most settings, customizing offerings to coarser segments is more meaningful than personalizing to individual customers simply because firms lack sufficient data for each customer. For example, in the sample dataset we use for our case study (Section 6) on eBay, we see that customers often interact with less than 5 items, of eBay's massive online catalog consisting of more than 4M items.

[1] We use the terms "segmentation" and "clustering" interchangeably.

The biggest challenge to carrying out segmentation is precisely this data sparsity. This challenge has become even more severe with firms being able to collect increasingly fine-grained observations such as direct purchases, ratings, and clicks, in addition to any demographic data such as age, gender, income, etc. These data are not only "big" (consisting of millions of customers and items), but also "complicated" in that they are (a) *unstructured*, with the items lacking well-defined feature information, (b) *diverse*, including actions that are represented on different scales (a click is not the same as a purchase is not the same as a rating), and (c) *highly sparse*, spanning only a small fraction of the entire item universe. Going back to the example above, eBay has a large and diverse product catalog consisting of products ranging from a Fitbit tracker/iPhone (products with well-defined attributes) to obscure antiques and collectibles (that lack any reasonable feature structure). Of these, each customer may click, purchase, or rate only a few items.

In this paper, we revisit the problem of segmentation but in the context of "big" and "complicated" data. These data characteristics pose new and unique challenges. First, traditional techniques within marketing don't apply. They assume that both customers and items have well-defined and consistent feature representations and often analyze small samples of customer populations. But, when the item universe is large and unstructured, customers can only be represented as large vectors with millions of entries, where each entry captures an action (say, purchase) taken on an item. These representations 'as is' are often meaningless for the purposes of segmentation. Almost all of their entries are missing and the lack of consistent feature representations of items means that missing entries can't be meaningfully imputed—for instance, a customer's purchase of an iPhone may reveal nothing about her propensity to purchase a particular antique. Existing techniques also become computationally intractable when classifying large populations of customers into segments. Second, the diversity of the types of actions captured in the vectors and their incompleteness make it difficult to assess similarity of customers. If a customer has clicked an item but another has purchased it, are they similar? How about customers who have purchased completely different subsets of items? This difficulty in obtaining a meaningful similarity measure precludes the application of standard clustering techniques in machine learning, despite being able to scale to large datasets.

To overcome the above challenges, we propose a *model-based embedding technique* that extends extant clustering techniques in machine learning to handle categorical observations from diverse data sources and having (many) missing entries. We focus on the setting where the objective of segmentation is to improve the performance on a prediction task. The precise prediction task depends on the application at hand, and includes predicting the probability of a customer clicking, purchasing, or liking an item. The algorithm takes as inputs the observations from a large population of customers and a probabilistic model class describing how the observations are generated from an individual customer. The choice of the model class is determined by the corresponding prediction task, as described below, and provides a systematic way to incorporate domain knowledge by leveraging the existing literature in marketing, which has proposed rich models describing individual customer behavior. It outputs an *embedding* for each customer—a vector representation in a low-dimensional Euclidean space whose dimension is much smaller than the number of items in the universe. The vector representations are then clustered, using a standard technique such as k-means, spectral clustering, mean-shift clustering, etc., to obtain the corresponding segments.

Put together, the algorithm proceeds in two sequential steps: *embed* and *cluster*. The *embed* step addresses the issue of diversity of the observed signals by first transforming the categorical observations into a continuous scale that makes different observations (such as purchases and ratings) comparable. It then deals with the issue of missing data by projecting the transformed observations onto a low-dimensional space, to obtain a vector representation for each customer. The *cluster* step then clusters the resulting vector representations to obtain the segments.

The key novelty of our algorithm is the *embed* step, which uses a probabilistic model to convert a categorical observation into its corresponding (log-)likelihood value under the model. For example, if a customer likes an item with probability $\alpha \in [0,1]$, then a "like" observation is transformed into $\log \alpha$ and "dislike" observation is transformed into $\log(1-\alpha)$. We call our algorithm *model-based* because it relies on a probabilistic model; Section 5 presents a case study in which we illustrate the choice of the model when the objective is to accurately predict if a new movie will be liked by a customer. We estimate the model parameters by pooling together the data from all customers and ignoring the possibility that different customers may have different model parameters. This results in a model that describes a 'pooled' customer—a virtual customer whose preferences reflect the aggregated preferences of the population. The likelihood transformations then measure how much a particular customer's preferences differ from those of the population's. The theoretical analysis in Section 3 shows that under reasonable assumptions, customers from different segments will have different (log-)likelihood values under the pooled model—allowing us to separate them out.

Our algorithm is inspired by existing ideas for clustering in the theoretical computer science literature. It systematically generalizes algorithms that are popular within the machine learning community. Particularly, when customer observations are continuous, a model is not necessarily needed to transform them into a comparable scale. Then, when there are no missing entries, our segmentation algorithm with the appropriate cluster step reduces to the standard spectral projection technique (Achlioptas and McSherry 2005, Kannan et al. 2005) for clustering real-valued observations. When there are missing entries, the embed step reduces to matrix factorization, which is commonly used in collaborative filtering applications (Koren et al. 2009).

Our work makes the following key contributions:

1. *Novel segmentation algorithm.* Our algorithm is designed to operate on large customer populations and large collections of unstructured items. Moreover, it is (a) *principled*, reducing to standard algorithms in machine learning in special cases; (b) *fast*, with an order of magnitude speedup compared to benchmark latent class models because it requires fitting only one model (as opposed to a mixture model); and (c) *flexible*, allowing practitioners to systematically incorporate problem-dependent structures through the model, providing a way to take advantage of rich literature in marketing proposing models for individual customer behavior.
2. *Analytical results.* Under a standard latent class model for customer observations, we derive necessary and sufficient conditions for exact recovery of the true segments. Specifically, we bound the asymptotic *misclassification rate*, defined as the expected fraction of customers incorrectly classified, of a nearest-neighbor classifier trained on customer embeddings obtained from the embed step in our algorithm. Given a universe of n items such that each customer provides at least $\log n$ observations, we show that the misclassification rate scales as $O\left(n^{-\frac{2\Lambda^2\alpha_{\min}^2}{81}}\right)$ where $0<\alpha_{\min},\Lambda<1$ are constants that depend on the underlying parameters of the model. In other words, when each customer provides $O(\log n)$ observations, our algorithm correctly classifies *all* customers into their respective segments as $n\to\infty$. Our results are similar in spirit to the conditions derived in existing literature for Gaussian mixture models (Kannan et al. 2005). However, existing proof techniques don't generalize to our setting. Our results are one of the first to provide such guarantees for latent class preference models.
3. *Empirical results.* We conducted three numerical studies to validate our methodology:
   (a) Using synthetic data, we show that our method obtains more accurate segments, while being upto 17× faster, than the standard latent class (LC) benchmark.
   (b) On the publicly available `MovieLens` dataset, we apply our segmentation method to solve the classical *cold-start problem*, which involves recommending new movies to users. We show that segmenting users using our method and customizing recommendations to each segment improves the recommendation accuracy by 48%, 55%, and 84% for drama, comedy, and action genres, respectively, when compared to a baseline method that treats all users as having the same preferences. It also outperforms the standard LC (by upto 13%) and empirical bayesian (by upto 19%) benchmarks used for capturing heterogeneity.
   (c) On a real-world dataset from eBay, we apply our segmentation methodology for personalizing similar product recommendations. We show that segmenting the population using our approach and customizing recommendations to each segment can result in upto 8% improvement in the recommendation quality, when compared to treating the population as homogeneous. The improvement of 8% is non-trivial because before our method, eBay tried several natural ways to segment (by similarity of demographics, frequency/recency of purchases, etc.), but the best of them resulted in $\sim 1\%$ improvement.

1.1. Relevant literature

Our work has connections to literature in both marketing and machine learning.

Marketing. Customer segmentation is a classical marketing problem, with origins dating back to the work of Smith (1956). Marketers classify various segmentation techniques into *a priori* versus *post hoc* and *descriptive* versus *predictive* methods, giving rise to a 2 × 2 classification matrix of these techniques (Wedel and Kamakura 2000). Our algorithm is closest to the post-hoc predictive methods, which identify customer segments on the basis of the estimated relationship between a dependent variable and a set of predictors. The traditional method for predictive clustering is automatic interaction detection (AID), which splits the customer population into non-overlapping groups that differ maximally according to a dependent variable, such as purchase behavior, on the basis of a set of independent variables, like socioeconomic and demographic characteristics (Assael 1970, Maclachlan and Johansson 1981). However, these approaches typically require large sample sizes to achieve satisfactory results. Ogawa (1987) and Kamakura (1988) proposed hierarchical segmentation techniques tailored to conjoint analysis, which group customers such that the accuracy with which preferences/choices are predicted from product attributes or profiles is maximized. These methods estimate parameters at the individual-level, and therefore are restricted by the no. of observations available for each customer. Clusterwise regression methods (Wedel and Kistemaker 1989, Wedel and Steenkamp 1989) overcome this limitation, as they cluster customers such that the regression fit is optimized within each cluster.

Latent class (or mixture) methods offer a statistical approach to the segmentation problem, and belong to two types: mixture regression and mixture multidimensional scaling models. Mixture regression models (Wedel and DeSarbo 1994) simultaneously group subjects into unobserved segments and estimate a regression model within each segment, and were pioneered by Kamakura and Russell (1989) who propose a clusterwise logit model to segment households based on brand preferences and price sensitivities. This was extended by Gupta and Chintagunta (1994) who incorporated demographic variables and Kamakura et al. (1996) who incorporated differences in customer choice-making processes, resulting in models that produce identifiable and actionable segments. Mixture multidimensional scaling (MDS) models (DeSarbo et al. 1994) simultaneously estimate market segments as well as preference structures of customers in each segment, for instance, a brand map depicting the positions of the different brands on a set of unobserved dimensions assumed to influence perceptual or preference judgments of customers.

The purpose of the above model-based approaches to segmenting customers is fundamentally different from our approach. These methods focus on characterizing the market segments in terms of product and customer features (such as prices, brands, demographics, etc.) by analyzing structured products (i.e. having well-defined attributes) and small samples of customer populations; consequently, they do not scale to directly classifying a large population of customers. Our algorithm is explicitly designed to classify the entire customer population into segments, and can be applied even when the data is less-structured or unstructured (refer to the case study in Section 6). Another distinction is that we can provide necessary and sufficient conditions under which our algorithm *guarantees* asymptotic recovery of the true segments under a latent class model, which is unlike most prior work. In addition, our algorithm can still incorporate domain knowledge by leveraging the rich models describing customer behavior proposed in existing literature.

Machine Learning. Clustering is defined as the problem of partitioning data objects into groups, such that objects in the same group are similar, while objects in different groups are dissimilar. The literature on clustering is vast; see Xu and Wunsch (2005) and Jain (2010) for excellent reviews. The most popular type of clustering approaches specify a distance/similarity measure between data points and determine the segments by optimizing a merit function that captures the "quality" of any given clustering. The popular k-means (and its variants k-medians, k-medoids, etc.), hierarchical clustering (Rokach and Maimon 2005), and spectral clustering (Shi and Malik 2000, Ng et al. 2002) are notable examples (refer to Appendix F for more details and additional references). However, as mentioned earlier, the diversity and sparsity in observations make it challenging to construct a meaningful similarity measure and limit the direct applicability of such techniques for clustering the customer population. At the same time, any of these techniques can be used in the *cluster* step of our algorithm (see Section 2.3), which enables us to tap into this vast literature. In contrast to the above similarity-based clustering approaches, model-based clustering techniques (Fraley and Raftery 2002, Zhong and Ghosh 2003) assume that each cluster is associated with an underlying probabilistic model and different clusters differ on the parameters describing the model. They estimate a finite mixture model (McLachlan and Peel 2004) to the data and classify customers based on the posterior membership probabilities. Our segmentation approach is closer to these techniques. The key distinction however is that these techniques estimate the parameters for each mixture component and can suffer when the number of samples available is limited, resulting in inaccurate segment classifications (see Section 4). Our approach, on the other hand, fits only a single model, the pooled model, and therefore is more robust to sparsity in the observed customer labels and at the same time, achieves significant computational speedup.

Our work also has methodological connections to work in the theoretical computer science literature on learning mixture models. Specifically, our model-based embedding technique extends existing techniques for clustering real-valued observations with no missing entries (Achlioptas and McSherry 2005, Kannan et al. 2005) to handle diverse categorical observations having (many) missing entries. Finally, method-of-moment based techniques with strong theoretical guarantees have recently been proposed for learning specific mixture models (Hsu and Kakade 2013, Anandkumar et al. 2014). However, these approaches require the lower-order moments to possess specific structures, which do not hold in our setting.

2. Setup and Algorithmic Framework

Our goal is to segment a population $[m] \overset{\text{def}}{=} \{1, 2, \ldots, m\}$ of m customers comprised of a fixed but unknown number K of non-overlapping segments. To carry out the segmentation, we assume access to individual-level observations that capture differences among the segments. The observations may come from diverse sources—organically generated clicks or purchases during online customer visits; ratings provided on review websites (such as Yelp, TripAdvisor, etc.) or recommendation systems (such as Amazon, Netflix, etc.); purchase attitudes and preferences collected from a conjoint study; and demographics such as age, gender, income, education, etc. Such data are routinely collected by firms as customers interact through various touch points. Without loss of generality, we assume that all the observations are categorical—any continuous observations may be appropriately quantized. The data sources may be coarsely curated based on the specific application but we don't assume access to fine-grained feature information.

To deal with observations from diverse sources, we consider a unified representation where each observation is mapped to a categorical label for a particular "item" belonging to the universe $[n] \overset{\text{def}}{=} \{1, 2, \ldots, n\}$ of all items. We use the term "item" generically to mean different entities in different contexts. For example, when observations are product purchases, the items are products and the labels binary purchase/no-purchase signals. When observations are choices from a collection of offer sets (such as those collected in a choice-based conjoint study), the items are offer sets and labels the IDs of chosen products. Finally, when observations are ratings for movies, the items are movies and the labels star ratings. Therefore, our representation provides a compact and general way to capture diverse signals. We index a typical customer by i, item by j, and segment by k.

In practice, we observe labels for only a small subset of the items for each customer. Because the numbers of observations can widely differ across customers, we represent the observed labels using an edge-labeled bipartite graph $\mathcal{P}$, defined between the customers and the items. An edge (i, j) denotes that we have observed a label from customer i for item j, with the edge-label x_{ij} representing the observed label. We call this graph the *customer-item preference graph*. We let $\boldsymbol{x}_i$ denote the vector[2] of observations for customer i with $x_{ij} = \phi$ if the label for item j from customer i is unobserved/missing. Let $N(i)$ denote the set of items for which we have observations for customer i. It follows from our definitions that $N(i)$ also denotes the set of neighbors of the customer node i

[2] We use bold lower-case letters like $\boldsymbol{x}, \boldsymbol{y}$ etc. to represent vectors

in the bipartite graph $\mathcal{P}$ and the degree $d_i \overset{\text{def}}{=} |N(i)|$, the size of the set $N(i)$, denotes the number of observations for customer i. Note that the observations for each customer are typically highly incomplete and therefore, $d_i \ll n$ and the bipartite graph $\mathcal{P}$ is highly sparse.

We assume that a customer's observations are generated according to an underlying parametric model from a pre-specified model class $\mathcal{F}(\Omega) = \{f(\boldsymbol{x};\omega) : \omega \in \Omega\}$, where Ω is the space of latent preference parameters, $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \mathcal{B} := \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$ is the vector of item labels, and $f(\boldsymbol{x};\omega)$ is the probability of observing the item labels $\boldsymbol{x}$ from a customer with model parameter ω. Here, $\mathcal{X}_j$ is the domain of possible categorical labels for item j. When all the labels are binary, $\mathcal{X}_j = \{0,1\}$ and $\mathcal{B} = \{0,1\}^n$. The choice of the parametric class depends on the application context and the prediction task at hand. For instance, for the task of predicting whether a customer likes a movie or not, $\mathcal{F}$ can be chosen to be the binary logit model class; for the task of predicting movie ratings (say, on a 5-star scale), $\mathcal{F}$ can be the ordered logit model class; and for the task of predicting which item will be purchased, $\mathcal{F}$ can be the multinomial logit (MNL) model class. Depending on the application, other models proposed within the marketing literature may be used. We provide concrete illustrations as part of case studies in the `MovieLens` dataset (Section 5) and eBay dataset (Section 6).

In order to segment the customer population, we assume that the population is heterogeneous, comprising different segments. Each segment is distinguished by its latent preference parameters, so that a population consisting of K segments is described by K distinct models $f_1, f_2, \cdots, f_K$ with corresponding latent preference parameters $\omega_1, \omega_2, \cdots, \omega_K$, respectively. Customer i in segment k generates the label vector $\boldsymbol{x}_i \sim f_k$, and we observe the labels x_{ij} for all the items $j \in N(i)$, for some preference graph $\mathcal{P}$. For ease of notation, we drop the explicit dependence of models in $\mathcal{F}$ on the parameter ω in the remainder of the discussion. Let $\boldsymbol{x}_i^{\text{obs}} \overset{\text{def}}{=} (x_{ij})_{j \in N(i)}$ denote the observed label vector from customer i and define the domain $\mathcal{B}^{(i)} = \left\{(x_j)_{j \in N(i)} \mid \boldsymbol{x} \in \mathcal{B}\right\}$. Given any model $f \in \mathcal{F}$, we define $f^{(i)}(\boldsymbol{y}) \overset{\text{def}}{=} \sum_{\boldsymbol{x}_i^{\text{mis}}} f(\boldsymbol{y}, \boldsymbol{x}_i^{\text{mis}})$ for each $\boldsymbol{y} \in \mathcal{B}^{(i)}$, where $\boldsymbol{x}_i^{\text{mis}}$ represent the missing labels vector for customer i and the summation is over all feasible missing label vectors when given the observations $\boldsymbol{y}$. Observe that $f^{(i)}$ defines a distribution over $\mathcal{B}^{(i)}$. Finally, let $\left|\boldsymbol{x}_i^{\text{obs}}\right|$ denote the length of the vector $\boldsymbol{x}_i^{\text{obs}}$; we have $\left|\boldsymbol{x}_i^{\text{obs}}\right| = |N(i)|$.

Algorithm 1 Algorithmic framework for customer segmentation

```
1: Input: observed labels $\boldsymbol{x}_1^{\text{obs}}, \boldsymbol{x}_2^{\text{obs}}, \ldots, \boldsymbol{x}_m^{\text{obs}}$; model class $\mathcal{F}$; the number of segments $K$
2: $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) \leftarrow \text{Embed}(\boldsymbol{x}_1^{\text{obs}}, \boldsymbol{x}_2^{\text{obs}}, \ldots, \boldsymbol{x}_m^{\text{obs}}, \mathcal{F})$; $\boldsymbol{v}_i$ is customer $i$'s embedding
3: $(\hat{z}_1, \ldots, \hat{z}_m) \leftarrow \text{Cluster}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m, K)$; $\hat{z}_i \in [K]$ is customer $i$'s segment label
4: Output: Segment labels $(\hat{z}_1, \ldots, \hat{z}_m)$
```

Under the above setup, we adopt the algorithmic framework presented in Algorithm 1 for segmenting the customers. The algorithm takes as inputs the observed customer labels, a model class $\mathcal{F}$ specifying how the labels are generated, and the number of segments, K, and outputs a clustering of the population into K segments. Our framework proceeds in two sequential steps. The first step *embeds* the customers into a low-dimensional Euclidean space, and the second step *clusters* the resulting embeddings to obtain the underlying segments. Our key contribution is the embedding algorithm, and therefore, we focus this section on describing the embedding algorithm. Once the embeddings are obtained, any of the existing algorithms can be used to cluster the embeddings—we provide specific recommendations towards the end of the section.

We describe two variants of our embedding algorithm. The first variant assumes that the model class $\mathcal{F}$ is fully-specified—that is, for a given value of $\omega \in \Omega$, the model $f(\cdot;\omega)$ completely specifies how an observation vector $\boldsymbol{x}$ is generated. The second, more general, variant allows the model class to be only partially specified.

2.1. Embedding algorithm for fully specified model class

For ease of exposition, we describe the algorithm separately for the cases with equal and unequal customer degrees in the preference graph $\mathcal{P}$. We start with the equal degree case and then deal with the more general unequal degree case.

2.1.1. Equal customer degrees. In this case, all customers have the same number of observations. The algorithm takes the observations in the form of the preference graph $\mathcal{P}$ and the model class $\mathcal{F}$ as inputs and outputs a uni-dimensional embedding of each customer. It proceeds as follows. Starting with the hypothesis that the population of customers is in fact homogeneous, it looks for evidence of heterogeneity to refute the hypothesis. Under the homogeneity hypothesis, it follows that the m observations $\boldsymbol{x}_1^{\text{obs}}, \boldsymbol{x}_2^{\text{obs}}, \dots, \boldsymbol{x}_m^{\text{obs}}$ are i.i.d. samples generated according to a single model in $\mathcal{F}$. Therefore, the algorithm estimates the parameters of a 'pooled' model $f_{\text{pool}} \in \mathcal{F}$ by pooling together all the observations and using a standard technique such as the maximum likelihood estimation (MLE) method. As a concrete example, consider the task of predicting whether a segment of customers likes a movie or not, so that $\mathcal{F}$ is chosen to be the binary logit, or logistic regression, model class where movie j is liked with probability $e^{\omega_j}/(1+e^{\omega_j})$, independent of the other movies. Then, the parameters of the pooled model can be estimated by solving the following MLE problem: $\max_{\omega_1,\omega_2,\dots,\omega_n} \sum_{i=1}^{m} \sum_{j \in N(i)} \log\left(\frac{e^{\mathbf{1}[x_{ij}=+1]\cdot\omega_j}}{1+e^{\omega_j}}\right)$,
where $x_{ij} = +1$ if customer i likes movie j and -1 if dislike. Because the objective function is separable, the optimal solution can be shown to be given by $\hat{\omega}_j = \log\left(\frac{\sum_{i:j\in N(i)} \mathbf{1}[x_{ij}=+1]}{\sum_{i:j\in N(i)} \mathbf{1}[x_{ij}=-1]}\right)$.

Once the pooled model is estimated, the algorithm assesses if the hypothesis holds by checking how well the pooled model explains the observed customer labels. Specifically, it quantifies the model fit by computing the (normalized) negative log-likelihood of observing $\boldsymbol{x}_i^{\text{obs}}$ under the pooled model, i.e., $v_i \overset{\text{def}}{=} \frac{1}{d_i} \cdot \left(-\log f_{\text{pool}}^{(i)}(\boldsymbol{x}_i^{\text{obs}})\right)$. A large value of v_i indicates that the observation $\boldsymbol{x}_i^{\text{obs}}$ is not well explained by the pooled model or that customer i's preferences are "far away" from that of the population. The value v_i is a uni-dimensional representation or embedding of customer i in the Euclidean space. We term it the *model-based embedding score* of the customer because it is obtained by transforming the observations $\boldsymbol{x}_i^{\text{obs}}$ into a real number by means of a model. The entire process is summarized in Algorithm 2.

Algorithm 2 Embedding algorithm with degree normalization

```
1: **Input:** observed labels $\boldsymbol{x}_1^{\text{obs}}, \boldsymbol{x}_2^{\text{obs}}, \ldots, \boldsymbol{x}_m^{\text{obs}}$ where $\left|\boldsymbol{x}_i^{\text{obs}}\right| = d_i \; \forall \; i$, model class $\mathcal{F}$
2: $f_{\text{pool}} \leftarrow$ estimated pooled model in $\mathcal{F}$
3: For each customer $i$ with observation $\boldsymbol{x}_i^{\text{obs}}$, the embedding $v_i \leftarrow \frac{1}{d_i} \cdot \left(-\log f_{\text{pool}}^{(i)}(\boldsymbol{x}_i^{\text{obs}})\right)$
4: **Output:** $\{v_1, v_2, \ldots, v_m\}$
```

We make the following remarks. First, our embedding algorithm is inspired by the classical statistical technique of analysis-of-variance (ANOVA), which tests the hypothesis of whether a collection of samples are generated from the same underlying population or not. For that, the test starts with the null hypothesis that there is no heterogeneity, fits a single model by pooling together all the data, and then computes the likelihood of the observations under the pooled model. If the likelihood is low (i.e., below a threshold), the test rejects the null hypothesis and concludes that the samples come from different populations. Our algorithm essentially separates customers based on the heterogeneity within these likelihood values.

Second, to understand why our algorithm should be able to separate the segments, consider the following simple case. Suppose a customer from segment k likes any item j with probability $f_k(\text{like}) = \alpha_k$ and dislikes it with probability $f_k(\text{dislike}) = 1 - \alpha_k$ for some $\alpha_k \in [0,1]$. Different segments differ on the value of the parameter α_k. Suppose q_k denotes the size of segment k, where $\sum_k q_k = 1$ and $q_k > 0$ for all k. Now, when we pool together a large number of observations from these customers, we should essentially observe that the population as a whole likes an item with probability $f_{\text{pool}}(\text{like}) \overset{\text{def}}{=} \sum_k q_k \alpha_k$; this corresponds to the pooled model. Under the pooled model, we obtain the embedding score for customer i as $\frac{1}{|N(i)|} \sum_{j \in N(i)} -\log f_{\text{pool}}(x_{ij})$ where each $x_{ij} \in \{\text{like}, \text{dislike}\}$. Now assuming that $|N(i)|$ is large and because the x_{ij}'s are randomly generated, the embedding score should concentrate around the expectation $\mathbb{E}_{\boldsymbol{X}_i \sim f_k}[-\log f_{\text{pool}}(\boldsymbol{X}_i)]$ where the random variable $\boldsymbol{X}_i$ takes value "like" with probability α_k and "dislike" with probability $1 - \alpha_k$, when customer i belongs to segment k. The value $\mathbb{E}_{\boldsymbol{X}_i \sim f_k}[-\log f_{\text{pool}}(\boldsymbol{X}_i)]$ is the cross-entropy between the distributions f_k

and f_{pool}. Therefore, if the cross-entropies for the different segments are different, our algorithm should be able to separate the segments[3]. We formalize and generalize these arguments in Section 3.

Third, our algorithm fits only one model—the 'pooled' model—unlike a classical latent class approach that fits, typically using the expectation-maximization (EM) method, a mixture distribution $g(\boldsymbol{x}) = \sum_k q_k f_k(\boldsymbol{x})$, where all customers in segment k are described by model $f_k \in \mathcal{F}$ and q_k represents the size (or proportion) of segment k. This affords our algorithm two advantages: (a) *speed:* up to 17× faster than the latent class benchmark (see Section 4) without the issues of initialization and convergence that are typical of EM-methods; and (b) *flexibility:* allows for fitting models from complex parametric classes $\mathcal{F}$ that more closely explain customer observations.

2.1.2. Unequal customer degrees. We now generalize our embedding algorithm to the case when customers may have unequal degrees in the preference graph $\mathcal{P}$. The issue here is that of normalization—the log-likelihood values $-\log f^{(i)}_{\text{pool}}(\boldsymbol{x}^{\text{obs}}_i)$ depend on the number of observations d_i and should be appropriately normalized in order to be meaningfully compared across customers. It is natural to normalize the log-likelihood values by the corresponding degrees, resulting in Algorithm 2 but applied to the unequal degree setting. Such degree normalization is appropriate when the observations across items are independent, so that the pooled distribution $f_{\text{pool}}(\boldsymbol{x})$ has a product form $f_{\text{pool},1}(x_1) \cdot f_{\text{pool},2}(x_2) \cdots f_{\text{pool},n}(x_n)$. In this case, the log-likelihood under the pooled model becomes $\log f^{(i)}_{\text{pool}}(\boldsymbol{x}^{\text{obs}}_i) = \sum_{j \in N(i)} \log f_{\text{pool},j}(x_{ij})$, which scales in the number of observations d_i.

Degree normalization, however, does not account for dependence structures that may be present in the item labels. For instance, in the extreme case when the observations across all items are perfectly correlated, such that customers either like all items or dislike all items with probability 0.5 each, the log-likelihood value does not depend on the number of observations. Yet, degree normalization divides the value by the degree, unfairly penalizing customers with only a few observations. To address this issue, we use entropy normalization:

$$v_i = \frac{-\log f^{(i)}_{\text{pool}}(\boldsymbol{x}^{\text{obs}}_i)}{H(f^{(i)}_{\text{pool}})} = \frac{-\log f^{(i)}_{\text{pool}}(\boldsymbol{x}^{\text{obs}}_i)}{-\sum_{\boldsymbol{y} \in \mathcal{B}^{(i)}} f^{(i)}_{\text{pool}}(\boldsymbol{y}) \log f^{(i)}_{\text{pool}}(\boldsymbol{y})} \tag{1}$$

where $H(f^{(i)}_{\text{pool}})$ denotes the *entropy* of distribution $f^{(i)}_{\text{pool}}$. When the observations across items are i.i.d., it can be seen that entropy normalization reduces to degree-normalization, upto constant factors. In addition, if the population has homogeneous preferences, all the customer embeddings concentrate around the value 1 (see Section 3.1). Therefore, deviations of embeddings from 1 indicates heterogeneity in customer preferences, allowing us to separate the different segments.

[3] Note that the cross-entropy is **not** a distance measure between distributions unlike the standard KL (Kullback-Leibler) divergence. Consequently, even when $f_k = f_{\text{pool}}$ for some segment k, the cross-entropy is not zero. Our algorithm relies on the cross-entropies being distinct to recover the underlying segments

Entropy normalizations have been commonly used in the literature for normalizing mutual information (Strehl and Ghosh 2002)—our normalization is inspired by that. In addition to accounting for dependency structures within the pooled distribution, it has the effect of weighting each observation by the strength of the evidence it provides. Because the log-likelihood value $-\log f_{\text{pool}}^{(i)}(\boldsymbol{x}_i^{\text{obs}})$ only provides incomplete evidence of how well f_{pool} captures the preferences of customer i when there are missing observations, we assess the confidence in the evidence by dividing the log-likelihood value by the corresponding entropy $H(f_{\text{pool}}^{(i)})$ of the distribution $f_{\text{pool}}^{(i)}$. Higher values of entropy imply lower confidence. Therefore, when entropy is high, the embedding score is low, indicating that there is insufficient evidence to conclude that customer i's observations are not well-explained by f_{pool}. Algorithm 3 summarizes entropy normalization.

Algorithm 3 Embedding algorithm with entropy normalization

```
  *same as Algorithm 2 except replace step 3 with*:
  Compute the embedding $v_i$ of customer $i$ via equation (1)
```

Entropy may be difficult to compute because, in general, it requires summing over an exponentially large space. For such cases, either the entropy computation may be approximated using existing techniques for (approximate) inference in probabilistic graphical models (Wainwright et al. 2008) or degree normalization of Algorithm 2 may be used as an approximation.

2.2. Embedding algorithm for partially specified model class

The discussion so far assumed that the model class $\mathcal{F}$ is fully-specified with $f(\cdot;\omega)$ specifying the complete joint distribution of each observation $\boldsymbol{x}$. In many practical settings, specifying the complete joint distribution structure is difficult especially when the item universe is large (for instance, millions in our eBay case study) and there are complex cross-effects among items, such as the correlation between the rating and purchase signal for the same item or complementarity effects among products from related categories (clothing, shoes, accessories, etc.). To handle such situations, we extend our embedding algorithm to the case when the model is only partially specified.

The precise setting we consider is as follows. The universe $[n]$ of items is partitioned into $B > 1$ "categories" $\{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_B\}$ such that $\mathcal{I}_b$ is the set of items in category $b \in [B] \overset{\text{def}}{=} \{1, 2, \ldots, B\}$, containing n_b items. A model describing the observations within each category is specified, but any interactions across categories are left unspecified. We let $\mathcal{F}_b(\Omega_b) = \{f(\boldsymbol{x}_b;\omega) : \omega \in \Omega_b\}$ denote the model class for category b, so that segment k is characterized by the B models $(f_{k1}, f_{k2}, \ldots, f_{kB})$ with $f_{kb} \in \mathcal{F}_b$ for all $1 \le b \le B$. Further, $\boldsymbol{x}_{ib}^{\text{obs}}$ denotes the vector of observations of customer i for items in category b; if there are no observations, we set $\boldsymbol{x}_{ib}^{\text{obs}} = \phi$.

Under this setup, we run our embedding algorithm (Algorithm 2 or 3) separately for each category of items. This results in a B-dimensional vector $\boldsymbol{v}_i = (v_{i1}, v_{i2}, \ldots, v_{iB})$ for each customer i, where v_{ib} is the embedding score computed by our algorithm for customer i and category b. When $\boldsymbol{x}_{ib}^{\text{obs}} = \phi$, we set $v_{ib} = \phi$. These vectors are compactly represented as the following $m \times B$ matrix with row i corresponding to customer i:

$$\mathcal{V} = \begin{bmatrix} v_{11} & v_{12} & \ldots & v_{1B} \\ v_{21} & v_{22} & \ldots & v_{2B} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m1} & v_{m2} & \ldots & v_{mB} \end{bmatrix}$$

When matrix $\mathcal{V}$ is complete, the algorithm stops and outputs $\mathcal{V}$ with $\boldsymbol{v}_i$ representing the embedding vector for customer i. Instead, if there are missing entries, we obtain the embeddings for the customers using low-rank matrix decomposition (or factorization) techniques, similar to those adopted in collaborative filtering applications (Koren et al. 2009). These techniques assume that the matrix $\mathcal{V}$ with missing entries can be factorized into a product of two low-rank matrices—one specifying the customer representation and the other a category representation, in a low-dimensional space. The low-rank structure naturally arises from assuming that only a small number of (latent) factors influence the cross-effects across categories. With this assumption, we compute an r-dimensional representation $\boldsymbol{u}_i \in \mathbb{R}^r$ for each customer i and $\boldsymbol{y}_b \in \mathbb{R}^r$ for each item category b by solving the following optimization problem:

$$\min_{U,Y} \sum_{i=1}^{m} \sum_{b=1}^{B} \mathbf{1}[v_{ib} \neq \phi] \cdot \left(v_{ib} - \boldsymbol{u}_i^\top \boldsymbol{y}_b\right)^2 \tag{2}$$

where row i of matrix $U \in \mathbb{R}^{m \times r}$ is $\boldsymbol{u}_i$ and row b of matrix $Y \in \mathbb{R}^{B \times r}$ is $\boldsymbol{y}_b$. Note that the rank $r \ll \min(m, B)$. When the no. of customers m or categories B is large, computing the low-rank decomposition may be difficult. But we can leverage recent work proposing scalable techniques for such matrices (as in collaborative filtering applications like Netflix), see the works of Takács et al. (2009), Mazumder et al. (2010) and references therein. Algorithm 4 summarizes the above process.

2.3. Clustering the embeddings to obtain segments

The final step is to cluster the customer embeddings to obtain the different segments. This step requires the choice of a clustering algorithm and the number of segments, K.

Clustering algorithm. Since the embeddings are vectors in the Euclidean space, any of the existing techniques designed for clustering real-valued vectors may be used. Popular candidates include k-means, mean-shift, kernel k-means, spectral clustering and spectral projection techniques. The choice of the technique depends on the specific application context and the corresponding strengths and weaknesses of the different clustering techniques. The k-means algorithm is the

Algorithm 4 Embedding algorithm for a partially specified model class $\mathcal{F}$

```
Input: observed labels $\boldsymbol{x}_1^{\text{obs}}, \boldsymbol{x}_2^{\text{obs}}, \ldots, \boldsymbol{x}_m^{\text{obs}}$, item partitioning $\{\mathcal{I}_1, \ldots, \mathcal{I}_B\}$, model class $\mathcal{F}_b$ for each $1 \leq b \leq B$, the rank $r \ll \min(m, B)$ of low-rank representation
$f_{\text{pool},b} \leftarrow$ estimated pooled model in $\mathcal{F}_b$ for all $1 \leq b \leq B$
Compute $v_{ib}$ for each customer $i$ and category $b$ using Algorithm 2 or 3 whenever $\boldsymbol{x}_{ib}^{\text{obs}} \neq \phi$ otherwise set $v_{ib} = \phi$
Create the $m \times B$ matrix $\mathcal{V}$ with row $i$ given by the vector $\boldsymbol{v}_i = (v_{i1}, v_{i2}, \ldots, v_{iB})$
If $\mathcal{V}$ is incomplete, compute rank $r$-factorization $\mathcal{V} \approx UY^\top$ where $U \in \mathbb{R}^{m \times r}, Y \in \mathbb{R}^{B \times r}$ by solving the optimization problem (2)
Output: $\mathcal{V}$ if it is complete and $UY^\top$ otherwise
```

most popular candidate. It iteratively chooses a set of centroids and partitions the data points by assigning each point to its closest centroid. It is widely used in practice and can scale to large numbers of samples. When the embedding is uni-dimensional, the mean-shift algorithm (Comaniciu and Meer 2002) may be used. It can identify clusters of arbitrary shape by means of an appropriately chosen kernel and also automatically determines the number of clusters. Spectral clustering (Ng et al. 2002) and spectral projection techniques (Kannan et al. 2005) are preferred candidates when the embeddings are "large" dimensional because they first project the vectors to a low-dimensional space before clustering. Finally, hierarchical clustering (Jain 2010) algorithms may be used if a particular application calls for nested clusters.

Number of segments. Any of the numerous techniques proposed in the literature including cross-validation (Wang 2010) and information-theoretic measures like the AIC, BIC, etc. (McLachlan and Peel 2004) can be used to pick the appropriate number of clusters K. When the embeddings are uni-dimensional, our technique also provides data-driven guidance on choosing a set of possible numbers of segments. Our theoretical analysis in Section 3 shows uni-dimensional embeddings of customers in the same segment concentrate around the same value, whereas those in different segments concentrate around distinct values, provided we have sufficient number of observations from each customer. Consequently, if we estimate the empirical distribution of the embeddings, using a general purpose technique like kernel density estimation (KDE), then the number of modes of the distribution should correspond to the underlying number of clusters[4]. Since the modes are sometimes difficult to distinguish, this approach provides us with a set of candidate numbers of segments. The precise number K may then be picked through cross-validation. But this approach does not scale to higher-dimensional embeddings.

[4] This is also the idea behind the mean-shift clustering algorithm which clusters data points by assigning them to the nearest mode of the KDE.

3. Theoretical Results

Our segmentation algorithm is analytically tractable and in this section, we derive theoretical conditions for how "separated" the underlying segments must be to guarantee asymptotic recovery using our algorithm. Our results are similar in spirit to existing theoretical results for clustering observations from mixture models, such as mixture of multivariate Gaussians (Kannan et al. 2005).

For the purposes of the theoretical analysis, we focus on the following standard setting—there is a population of m customers comprising K distinct segments such that a proportion q_k of the population belongs to segment k, for each $k \in [K] = \{1, 2, \ldots, K\}$. Segment k is described by distribution $\pi_k\colon \{-1,+1\}^n \to [0,1]$ over the domain $\mathcal{B} := \{-1,+1\}^n$. Note that this corresponds to the scenario when $\mathcal{X}_j = \{-1,+1\}$ for all items j (see the notation in Section 2). We frequently refer to $+1$ as *like* and -1 as *dislike* in the remainder of the section. Customer i's latent segment is denoted by $z_i \in [K]$, so that if $z_i = k$, then i samples a vector $\boldsymbol{x}_i \in \mathcal{B}$ according to distribution π_k, and then assigns the label x_{ij} for item j. We focus on asymptotic recovery of the true segment labels $\boldsymbol{z} = (z_1, z_2, \ldots, z_m)$, as the number of items $n \to \infty$.

The performance of our algorithm depends on the separation among the hyper-parameters describing the segment distributions π_k, as well as the number of data points available per customer. Therefore, we assume that the segment distributions are "well-separated" (the precise technical conditions are described below) and the number of data points per customer goes to infinity as $n \to \infty$. The proofs of all statements are in the Appendix.

3.1. Fully specified model class: independent item preferences

We first consider the case where π_k belongs to a fully specified model class $\mathcal{F}(\Omega)$, such that customer labels across items are independent. More precisely, we have the following model:

Definition 1 (latent class independent (LC-IND) model). Each segment k is described by distribution $\pi_k\colon \{-1,+1\}^n \to [0,1]$ such that labels $\{x_j\}_{j\in[n]}$ are independent and identically distributed. Denote $\alpha_k = \Pr_{\boldsymbol{x}\sim\pi_k}[x_j = +1]$ for all items $j \in [n]$, i.e. α_k is the probability that a customer from segment k likes an item. Customer i in segment k samples vector $\tilde{\boldsymbol{x}}_i$ according to distribution π_k and provides label $\tilde{x}_{ij}$ for item j.

We assume that the segment parameters are bounded away from 0 and 1, i.e. there exists a constant $\alpha_{\min} > 0$ such that $0 < \alpha_{\min} \leq \alpha_k \leq 1 - \alpha_{\min} < 1$ for all segments $k \in [K]$. Further, let $H(\beta_1, \beta_2) = -\beta_1 \log \beta_2 - (1-\beta_1)\log(1-\beta_2)$ denote the *cross-entropy* between the Bernoulli distributions $\text{Ber}(\beta_1)$ and $\text{Ber}(\beta_2)$ and $H(\alpha) = -\alpha\log(\alpha) - (1-\alpha)\log(1-\alpha)$ denote the binary entropy function, where $0 \leq \alpha \leq 1$. Let $\boldsymbol{V}_i$ denote the uni-dimensional embedding score computed by Algorithm 3, note that it is a random variable under the above generative model.

Given the above, we derive necessary and sufficient conditions to guarantee (asymptotic) recovery of the true customer segments under the LC-IND model.

3.1.1. Necessary conditions for recovery of true segments. We first present an important result concerning the concentration of the customer embeddings computed by our algorithm.

Lemma 1 **(Concentration of embedding scores under LC-IND model)**. *Given a customer population $[m]$ and collection of items $[n]$, suppose that the preference graph $\mathcal{P}$ is ℓ-regular, i.e. $d_i = \ell$ for all customers $1 \le i \le m$. Define the quantity $\alpha_{\text{pool}} \overset{\text{def}}{=} \sum_{k=1}^{K} q_k \alpha_k$. Then given any $0 < \varepsilon < 1$, the embedding scores computed by Algorithm 3 are such that:*

$$\Pr\left[\left|\boldsymbol{V}_i - \frac{H(\alpha_{z_i}, \alpha_{\text{pool}})}{H(\alpha_{\text{pool}})}\right| > \varepsilon \frac{H(\alpha_{z_i}, \alpha_{\text{pool}})}{H(\alpha_{\text{pool}})}\right] \le 4\exp\left(\frac{-2\ell\varepsilon^2\alpha_{\min}^2}{81}\right) + 12\exp\left(\frac{-2m\cdot\ell\cdot\varepsilon^2\bar{\alpha}_{\text{pool}}^2 \log^2(1-\bar{\alpha}_{\text{pool}})}{81\cdot\left(1-\log(1-\bar{\alpha}_{\text{pool}})\right)^2}\right)$$

where $\bar{\alpha}_{\text{pool}} \overset{\text{def}}{=} \min(\alpha_{\text{pool}}, 1-\alpha_{\text{pool}})$. In other words, the embedding scores of customers in segment k concentrate around the ratio $\frac{H(\alpha_k, \alpha_{\text{pool}})}{H(\alpha_{\text{pool}})}$, with high probability as the number of observations from each customer $\ell \to \infty$.

Lemma 1 reveals the necessary conditions our algorithm requires to recover the true customer segments. To understand the result, first suppose that $\alpha_{\text{pool}} \neq (1/2)$. Then, the above result states that the model-based embedding scores of customers in segment k concentrate around $\frac{H(\alpha_k, \alpha_{\text{pool}})}{H(\alpha_{\text{pool}})}$ which is proportional to $-\alpha_k \log \frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}} - \log(1-\alpha_{\text{pool}})$. Consequently, we require that $\alpha_k \neq \alpha_{k'}$ whenever $k \neq k'$ to ensure that the embedding scores of customers in different segments concentrate around distinct values. The result also states that the embedding scores of customers with similar preferences (i.e. belonging to the same segment) are close to each other, i.e. concentrate around the same quantity, whereas the scores of customers with dissimilar preferences (i.e. belonging to different segments) are distinct from each other. For this reason, although it is not a priori clear, our segmentation algorithm is consistent with the classical notion of distance- or similarity-based clustering, which attempts to maximize intra-cluster similarity and inter-cluster dissimilarity. When $\alpha_{\text{pool}} = (1/2)$, it follows that $H(\alpha_k, \alpha_{\text{pool}}) = H(\alpha_{\text{pool}})$ for *any* $0 \le \alpha_k \le 1$, and therefore all the customer embedding scores concentrate around 1. In this scenario, our algorithm cannot separate the customers even when the parameters α_k of different segments are distinct. Note that $\alpha_{\text{pool}} = \sum_k q_k \alpha_k$, which is the probability that a random customer from the population likes an item. Therefore, when $\alpha_{\text{pool}} = (1/2)$, the population is indifferent in its preferences for the items, resulting in all the customers being equidistant from the pooled customer.

The above discussion leads to the following theorem:

Theorem 1 **(Necessary conditions for recovery of true segments under LC-IND model)**. *Under the setup of Lemma 1, the following conditions are necessary for recovery of the true customer segments:*

1. *All segment parameters are distinct, i.e. $\alpha_k \neq \alpha_{k'}$ whenever $k \neq k'$, and* 2. $\alpha_{\text{pool}} \neq \frac{1}{2}$.

It is easy to see that the first condition is necessary for *any* segmentation algorithm. We argue that the second condition, i.e. $\alpha_{\text{pool}} \neq \frac{1}{2}$, is also necessary for the standard latent class (LC) segmentation technique. Specifically, consider two segments such that $q_1 = q_2 = 0.5$ and let $\alpha_1 = 1, \alpha_2 = 0$. Then, it follows that $\alpha_{\text{pool}} = q_1\alpha_1 + q_2\alpha_2 = \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 0 = \frac{1}{2}$. Let us consider only a single item, i.e. $n = 1$. Then, under this parameter setting, all customers in segment 1 will give the label +1 and all customers in segment 2 will give label −1. Recall that the LC method estimates the model parameters by maximizing the log-likelihood of the observed labels, which in this case looks like:

$$\log \mathcal{L} = \frac{m}{2} \log\left(q_1\alpha_1 + q_2\alpha_2\right) + \frac{m}{2} \log\left(q_1 \cdot (1-\alpha_1) + q_2 \cdot (1-\alpha_2)\right)$$

Then it can be seen that the solution $\hat{q}_1 = \hat{q}_2 = 0.5$ and $\hat{\alpha}_1 = \hat{\alpha}_2 = 0.5$ achieves the optimal value of the above log-likelihood function, and therefore is a possible outcome recovered by the LC method. This shows that the condition $\alpha_{\text{pool}} \neq \frac{1}{2}$ is also necessary for the standard LC method.

We also note that our results can be extended to the case when $\mathcal{P}$ is not ℓ-regular but with additional notation.

3.1.2. Sufficient conditions for recovery of true segments. Having established the necessary conditions, we now analyze the asymptotic *misclassification rate*, defined as the expected fraction of customers incorrectly classified, of our algorithm. In particular, we consider the following nearest-neighbor (NN) classifier $\hat{\boldsymbol{I}}(\cdot)$, where customer i is classified as:

$$\hat{\boldsymbol{I}}(i) = \operatorname*{arg\,min}_{k=1,2,\ldots,K} \frac{|\boldsymbol{V}_i - H_k|}{H_k}$$

where $H_k \overset{\text{def}}{=} \frac{H(\alpha_k, \alpha_{\text{pool}})}{H(\alpha_{\text{pool}})}$. Note that $H_k > 0$ since $0 < \alpha_{\min} \le \alpha_k \le 1 - \alpha_{\min} < 1$, for all $k \in [K]$.

Given the necessary conditions established above and to ensure that we can uniquely identify the different segments, we assume that the segments are indexed such that $\alpha_1 < \alpha_2 < \ldots < \alpha_K$. Then, we can prove the following recoverability result:

THEOREM 2 **(Asymptotic recovery of true segments under LC-IND model)**. *Under the setup of Lemma 1, suppose* $0 < \alpha_{\min} \le \alpha_1 < \alpha_2 < \cdots < \alpha_K \le 1 - \alpha_{\min}$ *and* $\alpha_{\text{pool}} \neq \frac{1}{2}$. *Further, denote* $\lambda = \min_{k=1,2,\ldots,K-1}(\alpha_{k+1} - \alpha_k)$. *Given any* $0 < \delta < 1$, *suppose that*

$$\ell \ge \frac{648}{\lambda^2} \cdot \left(\frac{\log \alpha_{\min}}{\log(1-\alpha_{\min}) \cdot \alpha_{\min}}\right)^2 \cdot \frac{1}{\log^2 \frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}}} \cdot \log(16/\delta)$$

Then, it follows that

$$\frac{1}{m} \sum_{i=1}^{m} \Pr\left[\hat{\boldsymbol{I}}(i) \neq z_i\right] < \delta$$

Further, when $\ell = \log n$ *and* $m \ge \left(\frac{1 - \log(1-\bar{\alpha}_{\text{pool}})}{\log(1-\bar{\alpha}_{\text{pool}})}\right)^2$, *we have:*

$$\frac{1}{m} \sum_{i=1}^{m} \Pr\left[\hat{\boldsymbol{I}}(i) \neq z_i\right] = O\left(n^{-\frac{2\Lambda^2 \alpha_{\min}^2}{81}}\right) \quad \textit{where the constant } \Lambda \overset{\text{def}}{=} \frac{\lambda}{2} \cdot \left(\frac{\left|\log \frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}}\right|}{|\log \alpha_{\min}|}\right)$$

Theorem 2 provides an upper bound on the misclassification rate of our segmentation algorithm in recovering the true customer segments. The first observation is that as the number of labels from each customer $\ell \to \infty$, the misclassification rate of the NN classifier goes to zero. The result also allows us determine the number of samples ℓ needed per customer to guarantee an error rate δ. In particular, ℓ depends on three quantities:

1. $\frac{1}{\lambda^2}$ where λ is the minimum separation between the segment parameters. This is intuitive—the "closer" the segments are to each other (i.e. smaller value of λ), the more samples are required per customer to successfully identify the true segments.
2. $\frac{1}{\log^2 \frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}}}$ where recall that α_{pool} is the probability that a random customer from the population likes an item. If $\alpha_{\text{pool}} \approx \frac{1}{2}$, then $\log^2 \frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}} \approx 0$ so that we require a large number of samples per customer. As α_{pool} deviates from $\frac{1}{2}$, the quantity $\log^2 \frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}}$ increases, so fewer samples are sufficient. This also makes sense—when $\alpha_{\text{pool}} = (1/2)$, our algorithm cannot identify the underlying segments, and the farther α_{pool} is from $(1/2)$, the easier it is to recover the true segments.
3. $\alpha_{\min}$, where as $\alpha_{\min} \to 0$, the number of samples required diverges. Note that $\alpha_{\min}$ (resp. $1-\alpha_{\min}$) specifies a lower (upper) bound on the segment parameters α_k—a small value of $\alpha_{\min}$ indicates that there exists segments with values of α_k close to either 0 or 1; and since the number of samples required to reliably estimate α_k (resp. $1-\alpha_k$) grows as $\frac{1}{\alpha_k^2}$ (resp. $\frac{1}{(1-\alpha_k)^2}$), ℓ must diverge as $\alpha_{\min} \to 0$.

Our result shows that as long as each customer provides at least $\log n$ labels, the misclassification rate goes to zero, i.e. we can accurately recover the true segments with high probability, as the number of items $n \to \infty$. Although the number of labels required from each customer must go to infinity, it must only grow logarithmically in the number of items n. Further, this holds for *any* population size "large enough".

Note that the NN classifier above assumes access to the "true" normalized cross-entropies H_k. In practice, we use "empirical" NN classifiers, which replace H_k by the corresponding cluster centroids of the embedding scores. Lemma 1 guarantees the correctness of this approach under appropriate assumptions, because the embedding scores of segment k customers concentrate around H_k.

3.2. Partially specified model class: independent within-category item preferences

We can extend the results derived above to the case when the distributions π_k belong to a partially specified model class, as discussed in Section 2.2. Specifically, suppose that the item set $[n]$ is partitioned into $B > 1$ (disjoint) categories: $\mathcal{I}_1 \uplus \mathcal{I}_2 \cdots \uplus \mathcal{I}_B$. The preferences of customers vary across the different categories, specifically we consider the following generative model:

Definition 2 (Latent Class Independent Category (LC-IND-CAT) Model). Each segment k is described using distribution $\pi_k\colon \{-1,+1\}^n \to [0,1]$ such that labels $\left\{x_{j_b}\right\}_{j_b \in \mathcal{I}_b}$ for items within a single category $b \in [B]$ are independent and identically distributed; but labels for items in different categories can have arbitrary correlations. Let $\boldsymbol{\alpha}_k = (\alpha_{k1}, \alpha_{k2}, \cdots, \alpha_{kB})$ be such that $\Pr_{\boldsymbol{x}\sim\pi_k}[x_{j_b} = +1] = \alpha_{kb}$ for each item $j_b \in \mathcal{I}_b$ and each category $b \in [B]$. Customer i in segment k samples vector $\tilde{\boldsymbol{x}}_i$ according to distribution π_k and provides label $\tilde{x}_{ij}$ for each item j.

The above model is general and can be used to account for correlated item preferences (as opposed to independent preferences considered in Section 3.1). As a specific example, suppose that for each item, we have two customer observations available: whether the item was purchased or not, and a like/dislike rating (note that one of these can be missing). Clearly these two observations are correlated and we can capture this scenario in the LC-IND-CAT model as follows: there are two item "categories"—one representing the purchases and the other representing the ratings. In other words, we create two copies of each item and place one copy in each category. Then, we can specify a joint model over the item copies such that purchase decisions for different items are independent, like/dislike ratings for different items are also independent but the purchase decision and like/dislike rating for the *same* item are dependent on each other. Similar transformations can be performed if we have more observations per item or preferences are correlated for a group of items. Therefore, the above generative model is fairly broad and captures a wide variety of customer preference structures.

As done for LC-IND model, we assume that the underlying segment parameters are bounded away from 0 and 1, i.e. there exists constant $\alpha_{\min} > 0$ such that $0 < \alpha_{\min} \le \alpha_{kb} \le 1 - \alpha_{\min} < 1$ for all segments $k \in [K]$, and all item categories $b \in [B]$. Let d_{ib} be the number of observations for customer i in category b and let $\overrightarrow{\boldsymbol{V}_i}$ denote the embedding vector computed by Algorithm 4 for customer i, note that it is a B-dimensional random vector under the generative model above.

3.2.1. Necessary conditions for recovery of true segments. We first state an analogous concentration result for the customer embedding vectors computed by our algorithm.

Lemma 2 **(Concentration of embedding vectors under LC-IND-CAT model)**. *For a population of m customers and n items partitioned into $B > 1$ categories, suppose that the preference graph $\mathcal{P}$ is such that each customer labels exactly $\ell_b > 0$ items in category b, i.e. $d_{ib} = \ell_b$ for all $1 \le i \le m$. Define the quantities $\alpha_{b,\text{pool}} \overset{\text{def}}{=} \sum_{k=1}^{K} q_k \alpha_{kb}$ for each item category b, $\ell_{\min} \overset{\text{def}}{=} \min_{b\in[B]} \ell_b$, and $\hat{\alpha}_{\text{pool}} \overset{\text{def}}{=} \min_{b\in[B]} \bar{\alpha}_{b,\text{pool}}$ where $\bar{\alpha}_{b,\text{pool}} \overset{\text{def}}{=} \min\left(\alpha_{b,\text{pool}}, 1-\alpha_{b,\text{pool}}\right)$. Then given any $0 < \varepsilon < 1$, the embedding vectors computed by Algorithm 4 are such that:*

$$\Pr\left[\left\|\overrightarrow{\boldsymbol{V}_i} - \boldsymbol{H}_{z_i}\right\|_1 > \varepsilon \left\|\boldsymbol{H}_{z_i}\right\|_1\right] \le 4 \cdot B \cdot \exp\left(\frac{-2\ell_{\min}\varepsilon^2\alpha_{\min}^2}{81}\right) + 12 \cdot B \cdot \exp\left(\frac{-2m \cdot \ell_{\min} \cdot \varepsilon^2 \hat{\alpha}_{\text{pool}}^2 \log^2(1-\hat{\alpha}_{\text{pool}})}{81\left(1-\log(1-\hat{\alpha}_{\text{pool}})\right)^2}\right)$$

In the lemma above, $\boldsymbol{H}_k = (H_{k1}, H_{k2}, \cdots, H_{kB})$ is a B-dimensional vector such that $H_{kb} = \frac{H(\alpha_{kb}, \alpha_{b,\text{pool}})}{H(\alpha_{b,\text{pool}})}$ (recall notation from section 3.1) and $\|\cdot\|_1$ denotes the $\mathcal{L}_1$-norm. Lemma 2 implies the following necessary conditions:

Theorem 3 **(Necessary conditions for recovery of true segments under LC-IND-CAT model).** *Under the setup of Lemma 2, the following conditions are necessary for recovery of the true customer segments:*

1. $\alpha_{b,\text{pool}} \neq \frac{1}{2}$ *for some category* $b \in [B]$.
2. *Let* $B' = \left\{b \in [B] : \ \alpha_{b,\text{pool}} \neq \frac{1}{2}\right\}$ *and denote* $(\boldsymbol{\alpha}_k)_{b \in B'}$ *as the sub-vector consisting of components corresponding to item categories* B'. *Then* $(\boldsymbol{\alpha}_k)_{b \in B'} \neq (\boldsymbol{\alpha}_{k'})_{b \in B'}$ *whenever* $k \neq k'$.

Similar to the LC-IND case, $\alpha_{b,\text{pool}} = (1/2)$ for *all* item categories implies that the population is indifferent over items in all the categories. However, we require the population to have well-defined preferences for at least one category in order to be able to separate the segments. Further, since $H_{kb} \propto -\alpha_{kb} \log \frac{\alpha_{b,\text{pool}}}{1-\alpha_{b,\text{pool}}} - \log(1-\alpha_{b,\text{pool}})$, we need $\alpha_{kb} \neq \alpha_{k'b}$ for at least one item category b where $\alpha_{b,\text{pool}} \neq \frac{1}{2}$ to ensure that the vectors $\boldsymbol{H}_k$ and $\boldsymbol{H}_{k'}$ are distinct, for any two segments $k \neq k'$.

3.2.2. Sufficient conditions for recovery of true segments. As for the case of the LC-IND model, we consider another NN classifier to evaluate the asymptotic misclassification rate of our segmentation algorithm, where customer i is classified as:

$$\hat{\boldsymbol{I}}_2(i) = \operatorname*{arg\,min}_{k=1,2,\ldots,K} \frac{\left\| \overrightarrow{\boldsymbol{V}_i} - \boldsymbol{H}_k \right\|_1}{\|\boldsymbol{H}_k\|_1}$$

Given the above necessary conditions, we can prove the following recoverability result:

Theorem 4 **(Asymptotic recovery of true segments under LC-IND-CAT model).** *Suppose that the conditions in Theorem 3 are satisfied. Denote* $\boldsymbol{w} = (w_1, w_2, \cdots, w_B)$ *with* $w_b = \left|\log \frac{\alpha_{b,\text{pool}}}{1-\alpha_{b,\text{pool}}}\right|$ *and* $\gamma = \min_{k \neq k'} \|\boldsymbol{w} \odot (\boldsymbol{\alpha}_k - \boldsymbol{\alpha}_{k'})\|_1$ *where* $\odot$ *represents element-wise product. Under the setup of Lemma 2 and given any* $0 < \delta < 1$, *suppose that*

$$\ell_{\min} \geq \frac{648B^2}{\gamma^2} \cdot \left(\frac{\log \alpha_{\min}}{\log^2(1-\alpha_{\min}) \cdot \alpha_{\min}}\right)^2 \log(16B/\delta)$$

Then, it follows that

$$\frac{1}{m} \sum_{i=1}^{m} \Pr\left[\hat{\boldsymbol{I}}_2(i) \neq z_i\right] < \delta$$

Further, when $\ell_{\min} = \log n$ *and* $m \geq \left(\frac{1-\log(1-\hat{\alpha}_{\text{pool}})}{\log(1-\hat{\alpha}_{\text{pool}})}\right)^2$, *for fixed* B *we have:*

$$\frac{1}{m} \sum_{i=1}^{m} \Pr\left[\hat{\boldsymbol{I}}_2(i) \neq z_i\right] = O\left(n^{\frac{-2\Gamma^2 \alpha_{\min}^2}{81}}\right) \quad \textit{where the constant } \Gamma \overset{\text{def}}{=} \frac{\gamma}{2B} \cdot \left|\frac{\log(1-\alpha_{\min})}{\log \alpha_{\min}}\right|$$

We make a few remarks about Theorem 4. First, as $\ell_{\min} \to \infty$, i.e. the number of labels in each item category $\ell_b \to \infty$, the misclassification rate of the NN classifier goes to zero. Second, to achieve misclassification rate of at most δ, the number of samples $\ell_{\min}$ scales as

1. $\frac{1}{\gamma^2}$ where γ is the minimum weighted $\mathcal{L}_1$-norm of the difference between the parameter vectors of any two segments. This is similar to a standard "separation condition"—the underlying segment vectors $\boldsymbol{\alpha}_k$ should be sufficiently distinct from each other, as measured by the $\mathcal{L}_1$-norm. However, instead of the standard $\mathcal{L}_1$-norm, we require a weighted form of the norm, where the weight of each component is given by $w_b = \left|\log \frac{\alpha_{b,\text{pool}}}{1-\alpha_{b,\text{pool}}}\right|$. If $\alpha_{b,\text{pool}} \approx \frac{1}{2}$, then $w_b \approx 0$ so that the separation in dimension b is weighed lower than categories where $\alpha_{b,\text{pool}}$ is sufficiently distinct from $\frac{1}{2}$. This follows from the necessary condition in Theorem 3 and is a consequence of the simplicity of our algorithm that relies on measuring deviations of customers from the population preference.
2. B^2, this is expected—as the number of categories increases, we require more samples to achieve concentration in *all* dimensions of the embedding vector $\overrightarrow{\boldsymbol{V}_i}$.
3. $\alpha_{\min}$, the dependence on which is similar to the LC-IND model case, but with an extra factor of $\log^2(1-\alpha_{\min})$ in the denominator, indicating a more stronger dependence on $\alpha_{\min}$.

Finally, it follows that a logarithmic number of labels in *each* category is sufficient to guarantee recovery of the true segments with high probability as the total number of items $n \to \infty$, provided the population size m is "large enough".

4. Computational study: Accuracy of model-based embedding technique

The theoretical analysis above provides asymptotic recovery guarantees for our approach when all customers have equal degrees in the preference graph $\mathcal{P}$ and customer degrees grow to infinity. This section supplements our theoretical results with the results from a computational study that tests the performance of our technique when customer degrees are unequal and finite. The study analyzes the misclassification rate of our algorithm when provided synthetic observations generated on a preference graph $\mathcal{P}$ with fixed numbers of customers and items. Our results show that as the average customer degree decreases, the misclassification rate of our algorithm increases, as expected. However, the error increases at a much smaller rate compared to a standard latent class (LC) benchmark, which classifies customers using the estimated posterior membership probabilities in different segments (details below). Specifically, our results show that compared to the LC benchmark, our approach is (1) 28% more accurate in recovering the true customer segments and (2) faster, with average 17× speedup in computation time.

Setup. We chose $m = 2000$ customers and $n = 100$ items and considered the following standard latent class generative model: The population consists of K customer segments with q_k denoting the proportion of customers in segment k; we have $q_k > 0$ for all $k \in [K]$ and $\sum_{k=1}^{K} q_k = 1$. The preference graph follows the standard Erdős-Rényi (Gilbert) model with parameter $0 < p < 1$: each edge (i, j) between customer i and item j is added independently with probability p. The parameter $1-p$ quantifies the *sparsity* of the graph: higher the value of $1-p$, sparser the graph. All customers in segment $k \in [K]$ generate binary labels as follows: given parameter $\alpha_k \in (0, 1)$, they provide rating $+1$ to item j with probability α_k and rating -1 with probability $1-\alpha_k$.

We denote each ground-truth model type by the tuple: $(K, 1-p)$. We generated 15 models by varying K over the set $\{5, 7, 9\}$ and $1-p$ over the set $\{0, 0.2, 0.4, 0.6, 0.8\}$. For each value of K, we sampled the segment proportions from a Dirichlet distribution with parameters $\beta_1 = \beta_2 = \cdots = \beta_K = K+1$ which tries to ensure that all segments have sufficiently large sizes by placing a large mass on equal proportions. Similarly, for each K, the parameters α_k are chosen as $\alpha_k = 0.05 + 0.9(k-1)/(K-1)$ (K uniformly spaced points in the interval $[0.05, 0.95]$) for all $1 \leq k \leq K$.

For each ground-truth model type, we randomly generated 30 model instances as follows: (a) randomly partition the customer population into K segments with segment k having proportion q_k; (b) randomly generate the customer-item preference graph by adding edge (i, j) between customer i and item j with probability p; and (c) for each edge (i, j) in the preference graph, assign rating $+1$ with probability α_k and -1 with prob. $1-\alpha_k$ where customer i belongs to segment k.

LC benchmark. Given the preference graph $\mathcal{P}$ with the corresponding ratings $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m$, the LC method estimates the model parameters by solving the following MLE problem:

$$\max_{\substack{q_1, q_2, \cdots, q_K \\ \alpha_1, \cdots, \alpha_K}} \sum_{i=1}^{m} \log \left(\sum_{k=1}^{K} q_k \prod_{j \in N(i)} \alpha_k^{\mathbf{1}[x_{ij}=+1]} (1-\alpha_k)^{\mathbf{1}[x_{ij}=-1]} \right) \text{ s.t. } \sum_k q_k = 1, q_k \geq 0, 0 \leq \alpha_k \leq 1 \;\forall\; k$$

To ensure that the results for the LC benchmark are robust to how the parameters are estimated, we solved the above MLE problem using two different methods: (a) the popular EM algorithm and (b) Sequential Least Squares Programming (SLSQP)[5]—a standard off-the-shelf solver (see Appendix C for more details). After the model parameters are obtained, customers are assigned to the segment for which the posterior probability of membership is the largest. Note that the LC method estimates a total of $2 \cdot K$ parameters.

Model-based embedding algorithm. We computed the embedding scores of the customers using Algorithm 3. The pooled model f_{pool} is described by a single parameter, $\alpha_{\text{pool}} = \frac{\sum_{i=1}^{m} \sum_{j \in N(i)} \mathbf{1}[x_{ij}=+1]}{\sum_{i=1}^{m} |N(i)|}$. We clustered the embedding scores using the k-means algorithm to obtain the segments, and call our approach α-EMBED.

[5] Specifically, we used Python SciPy library's `minimize` interface: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.minimize.html#scipy.optimize.minimize`

Table 1 Percentage accuracy in recovering true segments for different parameter settings

K	$1-p$	LC		α-EMBED	% IMPROVEMENT		AVERAGE SPEEDUP (X)	
		EM	SLSQP		over EM	over SLSQP	over EM	over SLSQP
5	0.0	99.0	99.0	98.7	-0.3^{**}	-0.3^{**}	20	251
	0.2	98.1	97.9	97.5	-0.6^{**}	-0.4^{**}		
	0.4	96.2	96.0	95.1	-1.1^{**}	-0.9^{**}		
	0.6	91.8	91.3	89.2	-2.8^{**}	-2.3^{**}		
	0.8	75.7	79.2	75.9	0.3	-4.2^{**}		
7	0.0	92.9	92.7	92.1	-0.9^{**}	-0.6^{**}	16	216
	0.2	89.3	89.4	88.3	-1.1^{**}	-1.2^{**}		
	0.4	82.5	84.1	83.1	0.7	-1.2^{**}		
	0.6	71.9	72.2	74.6	3.8^{**}	3.3^{*}		
	0.8	54.2	49.2	61.4	13.3^{**}	24.8^{**}		
9	0.0	72.5	81.2	80.8	11.4^{**}	-0.5	15	324
	0.2	65.7	71.2	75.6	15.1^{**}	6.2^{**}		
	0.4	58.3	58.1	70.4	20.7^{**}	21.2^{**}		
	0.6	47.9	47.9	61.5	28.4^{**}	28.4^{**}		
	0.8	39.1	35.2	49.1	25.6^{**}	39.5^{**}		

The parameters are K—number of customer segments and $(1-p)$—sparsity of the preference graph. Each observation above is an average over 30 experimental runs. p-values computed according to a paired samples t-test.
**: $p < 0.01$, *: $p < 0.05$

Results and Discussion. We measure the quality of the recovered clusters in terms of *accuracy*, defined as $\text{Accuracy}^{\mathsf{algo}} = 100 \times \left(\frac{1}{m}\sum_{i=1}^{m} \mathbf{1}[\hat{z}_i^{\mathsf{algo}} = z_i]\right)$, where z_i is the true segment of customer i, $\hat{z}_i^{\mathsf{algo}}$ is the segment label assigned by method algo, and $\mathsf{algo} \in \{\text{LC}, \alpha\text{-EMBED}\}$. We label the true segments such that $\alpha_1 < \alpha_2 < \cdots < \alpha_K$. Then, for the LC method, we assign the segment labels in order of the estimated alpha parameters $\hat{\alpha}_k$, so that $\hat{\alpha}_1 < \hat{\alpha}_2 < \cdots < \hat{\alpha}_K$. For the α-EMBED method, recall from Lemma 1 that the embedding scores of customers in segment k concentrate around $H(\alpha_k, \alpha_{\text{pool}})/H(\alpha_{\text{pool}})$. Since $H(\alpha_k, \alpha_{\text{pool}}) = -\alpha_k \log \frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}} - \log(1-\alpha_{\text{pool}})$, it follows that $H(\alpha_k, \alpha_{\text{pool}})$ is either increasing or decreasing in α_k depending on whether $\alpha_{\text{pool}} < \frac{1}{2}$ or $> \frac{1}{2}$. Therefore, we assign the segment labels in the increasing (resp. decreasing) order of the customer embedding scores when $\alpha_{\text{pool}} < \frac{1}{2}$ (resp. $\alpha_{\text{pool}} > \frac{1}{2}$).

Table 1 reports the accuracy of the LC and α-EMBED methods. Since there is no model misspecification, the LC method is statistically optimal and we see that it is able to recover the true customer segments accurately when the preference graph is dense, but its performance suffers as the sparsity, $1-p$, increases. As sparsity increases, the number of data points per customer decreases, so the LC method encounters insufficient data to reliably estimate the $2K$ model parameters, particularly for larger values of K. The α-EMBED method, on the other hand, has comparable performance when there is enough data relative to the number of parameters being estimated. But it significantly outperforms the LC benchmark, by upto 28%, as the level of sparsity increases. The reason is that since the α-EMBED method estimates only a single parameter, it can make more efficient use of available data to determine the true segments. We also note that the performance improvement of α-EMBED over the LC benchmark is robust to the specific optimization method used for estimating the parameters of the LC model, be it EM or SLSQP.

Finally, as reported in Table 1, the fact that we estimate only a single model as opposed to K models results in an average 17× speedup compared to the EM method[6], which is also sensitive to the initialization of the model parameters. This speedup becomes more significant when we have millions of customers and items, such as in our case-study at eBay (see Section 6), where the LC method is too computationally expensive and becomes infeasible to implement in practice.

5. Case study 1: Cold start recommendations in MovieLens dataset

Using the popular `MovieLens` (Herlocker et al. 1999) dataset, we now illustrate how our segmentation methodology can be applied to solve the classical cold-start problem in recommendation systems. This problem involves recommending *new* movies to users[7]. It is challenging because, by definition, new movies do not have any existing ratings. Yet, we need to account for heterogeneity in user preferences and personalize the recommendations. The baseline approach assumes that the population has homogeneous preferences and recommends the same set of movies to all the users. Compared to this benchmark, we show that segmenting the user population using our approach and customizing the recommendations to each segment can result in upto 48%, 55% and 84% improvements in the recommendation quality for drama, comedy, and action movies, respectively. In addition, compared to standard benchmarks used for capturing heterogeneity, our method outperforms: (1) latent class (LC) method by 8%, 13% and 10% and (2) empirical bayesian (EB) technique by 19%, 12% and 8%, respectively for the three genres. In addition, we achieve 20× speedup in the computation time over the LC benchmark.

Data Processing. The `MovieLens` dataset consists of 1M movie ratings (on 1-5 scale) from $6,040$ users for $3,952$ movies. For our analysis, we choose the three genres with the most number of movies in the dataset—drama, comedy, and action. We pose the new movie recommendation task as the following prediction problem: given a movie, what is the probability that the user *likes* the movie? We say that a user *likes* a movie if the rating for the movie is greater than or equal to the average rating of the user across all the movies she rated, and *dislikes* the movie otherwise. Since the prediction task is only concerned with a binary (like/dislike) signal, we transform the raw user ratings to a binary like $(+1)$ and dislike (-1) scale. Therefore, the preference graph consists of users on the left, movies on the right and the binary like/dislike ratings representing the edges.

[6] SLSQP takes significantly longer to converge, resulting in an average 264× speedup

[7] To be consistent with the standard terminology in this problem context, we refer to customers as "users" in the remainder of the paper

We solve the prediction problem separately for each genre since user preferences can vary across different genres (see Appendix D.3 for the case when movies of different genres are considered together, where we also showcase the application of our technique when the model generating user ratings is only partially specified as in Section 2.2). Further, since our goal is to recommend new movies to users (which have no or only a few ratings in practice), we select movies, for each genre, that have been rated by atleast 30 users as part of the training set, and all others as part of the test set. Statistics for the training and test datasets are given in Table 2.

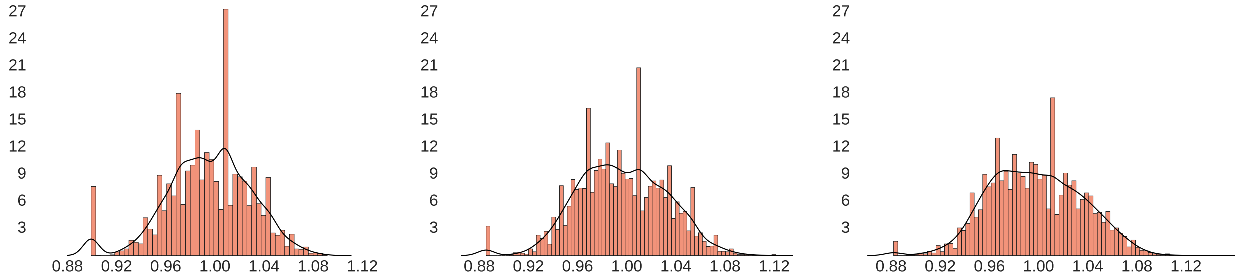
Methods. The cold-start problem (Schein et al. 2002) has been studied extensively in the recommendation systems literature with solutions utilizing user-level and item-level attributes (Park and Chu 2009, Zhang et al. 2014) as well as social connections such as Facebook friends/likes or Twitter followers (Lin et al. 2013, Sedhain et al. 2014). For our case study, the only information we use are the (transformed) user ratings and the genre of the movies. Consequently, existing collaborative filtering techniques are not directly applicable. In addition, the preference graph contains many missing ratings[8] and therefore, existing similarity-based clustering techniques that compute a similarity measure between users perform poorly (see the discussion at the end of the section). However, our segmentation approach is precisely designed to handle sparsity in user observations and make it a natural choice in this context.

Recall that the goal is to predict the probability that a user gives +1 rating to a movie. To determine the benefits of segmentation for solving this prediction problem, we contrast two approaches—(1) *population model* : the user population is homogeneous so that all users have the same probability α for liking any movie; and (2) *segmentation model* : the population is composed of K segments, such that users in segment k have probability α_k of liking any movie. We first estimate the model parameters in both approaches using the training data and then, based on the estimated parameters, predict the ratings given by each user for movies in the test set. Let U denote the set of all users, and $N^{\text{train}}(i)$ and $N^{\text{test}}(i)$ denote respectively the set of movies in the training and test set rated by user i. For the *population model* approach, which we call POP, the maximum likelihood estimate (MLE) for parameter α is obtained by pooling all the ratings: $\alpha_{\text{pool}} = \frac{\sum_{i \in U} \sum_{j \in N^{\text{train}}(i)} \mathbf{1}[r_{ij}=+1]}{\sum_{i \in U} \left| N^{\text{train}}(i) \right|}$ where r_{ij} is the rating given by user i for movie j. For the *segmentation model* approach, we use our model-based embedding technique α-EMBED, described earlier in Section 4, which computes user embeddings based on the estimated pooled model α_{pool} and then clusters the embeddings using k-means to obtain the segments. Then, we compute each segment parameter as $\alpha_k^{\text{EMBED}} = \frac{\sum_{i \in U} \mathbf{1}[\hat{z}_i^{\text{EMBED}}=k] \cdot \left(\sum_{j \in N^{\text{train}}(i)} \mathbf{1}[r_{ij}=+1] \right)}{\sum_{i \in U} \mathbf{1}[\hat{z}_i^{\text{EMBED}}=k] \cdot \left| N^{\text{train}}(i) \right|}$ where $\hat{z}_i^{\text{EMBED}} \in [K]$ represents the assigned segment label for user i.

Given the above, the prediction for user i and new movie j_{new} is carried out as follows:

[8] The average sparsity, i.e. # edges/ (# users · # movies) is ∼ 7%; see Table 2

Figure 1 Density of user embedding scores for different genres—**Left**: Action, **Center**: Comedy, **Right**: Drama



Note. The x-axis corresponds to the embedding scores and the y-axis represents the number of users.

Table 2 Training/test data statistics and aggregate rating prediction accuracy for the different genres

Genre	Train Data			Test Data		Accuracy				% Improvement over		
	Users	Movies	Ratings	Movies	Ratings	POP	Benchmarks		α-EMBED	POP	EB	LC
							EB	LC				
Action (K=2)	6012	453	257K	42	403	30.7	47.6	51.2	56.4	83.7	18.5	10.1
Comedy (K=4)	6031	945	354K	218	2456	37.7	52.4	51.8	58.4	54.9	11.5	12.7
Drama (K=4)	6037	1068	350K	425	4627	38.6	53.2	53.0	57.2	48.2	7.5	7.9

The number in parentheses represents the number of segments determined for each genre.

1. For the POP method, $\hat{r}^{\text{POP}}_{ij_{\text{new}}} = +1$ if $\alpha_{\text{pool}} \geq 0.5$, else $\hat{r}^{\text{POP}}_{ij_{\text{new}}} = -1$.
2. For the α-EMBED method, $\hat{r}^{\text{EMBED}}_{ij_{\text{new}}} = +1$ if $\alpha^{\text{EMBED}}_{\hat{k}} \geq 0.5$, else $\hat{r}^{\text{EMBED}}_{ij_{\text{new}}} = -1$ where $\hat{k} = \hat{z}^{\text{EMBED}}_i$.

We also compare our approach to two standard benchmarks that capture heterogeneity in user preferences: the LC method and the EB approach (Rossi et al. 2005) commonly used in the marketing literature. Refer to Appendix D.1 for more details.

There are many metrics to evaluate recommendation quality (Shani and Gunawardana 2011). Since we are dealing with binary ratings, a natural metric is *accuracy*, i.e. the fraction of ratings that are predicted correctly. More precisely, let U^{test} denote the set of all users in the test set, note that $U^{\text{test}} \subseteq U$. Then for each user $i \in U^{\text{test}}$, we compute the individual accuracy as: $\text{Accuracy}^{\mathsf{method}}_i = \frac{1}{|N^{\text{test}}(i)|} \sum_{j_{\text{new}} \in N^{\text{test}}(i)} \mathbf{1}[r_{ij_{\text{new}}} = \hat{r}^{\mathsf{method}}_{ij_{\text{new}}}]$ where $\mathsf{method} \in \{\text{POP}, \text{LC}, \text{EB}, \text{EMBED}\}$. The aggregate accuracy is then computed as: $\text{Accuracy}^{\mathsf{method}} = 100 \times \left(\frac{1}{|U^{\text{test}}|} \cdot \sum_{i \in U^{\text{test}}} \text{Accuracy}^{\mathsf{method}}_i \right)$. In the same manner, we can also compute the aggregate accuracy for a given segment k of users (identified by the α-EMBED method):

$$\text{Accuracy}^{\mathsf{method}}_k = 100 \times \left(\frac{1}{|\{i \in U^{\text{test}} : \hat{z}^{\text{EMBED}}_i = k\}|} \sum_{i \in U_{\text{test}} : \hat{z}^{\text{EMBED}}_i = k} \text{Accuracy}^{\mathsf{method}}_i \right)$$

Results and Discussion. Figure 1 shows the kernel density estimate of the user embeddings for each genre. As noted in Section 2.3, our approach provides data-driven guidance on choosing a

Table 3 Comparison of rating prediction accuracy of population model and our model-based embedding technique by individual user segments

Genre	α_{pool}	User Segments	α_k^{EMBED}	$\text{Accuracy}_k^{\text{POP}}$	$\text{Accuracy}_k^{\text{EMBED}}$	% increase
Action	0.537	Segment 1 (2900)	0.658	35.6	35.6	-
		Segment 2 (3112)	0.441	26.9	73.1	171.7
Comedy	0.543	Segment 1 (941)	0.749	45.2	45.2	-
		Segment 2 (2062)	0.631	45.9	45.9	-
		Segment 3 (1869)	0.495	33.2	66.8	101.3
		Segment 4 (1159)	0.360	26.4	73.6	178.7
Drama	0.545	Segment 1 (1164)	0.736	50.1	50.1	-
		Segment 2 (1975)	0.620	43.5	43.5	-
		Segment 3 (1769)	0.485	36.1	63.9	77.0
		Segment 4 (1129)	0.342	22.5	77.5	244.4

The numbers in parentheses represent the size of each user segment. % increase denotes the percentage improvement in accuracy of our segmentation approach over the population model

set of possible numbers of segments based on the number of modes in the estimated density; we try values of $K \in \{2,3,4,5\}$ and choose the one that maximizes accuracy on a *validation set*[9]—a subset of the training data consisting of movies that have relatively "small" number of ratings, i.e. at most 50 ratings. We also show that our approach is robust to the choice of K; see Appendix D.2 for details. After segmenting the users, we predicted their ratings for new movies as outlined above and compute the accuracy metrics for the different approaches.

Table 2 reports the aggregate accuracy for each of the genres. The benefits of segmentation can be seen across all the genres, with improvements upto 84% (for the action genre) in the prediction accuracies. The population model treats the preferences of all users as being the same and performs poorly since it ends up recommending the same set of movies to all the users. The segmentation model, on the other hand, makes different recommendations to the different user segments, and consequently performs significantly better. Further, using our segmentation algorithm performs better than both the LC (upto 13% for the comedy genre) and EB (upto 19% for the action genre) methods. This suggests that a discrete form of heterogeneity, resulting from a "hard" separation of the users into distinct segments, is better than a continuous or "soft" form of heterogeneity (as considered by the benchmarks) for the cold-start recommendation problem. In addition, our method is upto 20× faster than the LC method[10] when the population is grouped into $K = 4$ segments, again highlighting the fact that our algorithm is fast and can scale to large dimensions.

To understand where the accuracy improvements come from, Table 3 displays the accuracy of the POP and α-EMBED methods, broken down by individual user segments computed by the

[9] This is standard practice in the machine learning literature; see for instance Section 4.3 in Abu-Mostafa et al. (2012)

[10] We use the EM algorithm to estimate the model parameters.

α-EMBED method. Also shown are the estimated pooled model α_{pool} and segment parameters α_k^{EMBED}. Now observe that for segments 1 & 4 in the drama and comedy genres, the estimated parameters α_k^{EMBED} are furthest from the pooled parameter α_{pool}. In other words, these segments contain users whose preferences are very different from that of the population, i.e. *esoteric* preferences, which are not captured well by the pooled model α_{pool}. Using the segment parameters α_k^{EMBED} for the rating predictions results in significant improvements in the accuracy for segment 4 users—upto 1.8× and 2.5× increase for the comedy and drama genres respectively. Note that we do not observe any improvement for segment 1 users, this is because of our experimental setup which involves a coarse-grained rating prediction based on a threshold of 0.5 (see the setup above). The users in the intermediate segments 2 & 3, on the other hand, have preferences that are very similar to those of the population, i.e *mainstream* preferences. However, we are still able to distinguish between users in these segments resulting in improved rating prediction accuracy for segment 3 users. The improvements are lower than for the esoteric segments, since the pooled model is already able to capture the preferences of the mainstream users. The story is similar for the action genre where segment 1 (resp. 2) behaves as the mainstream (resp. esoteric) segment.

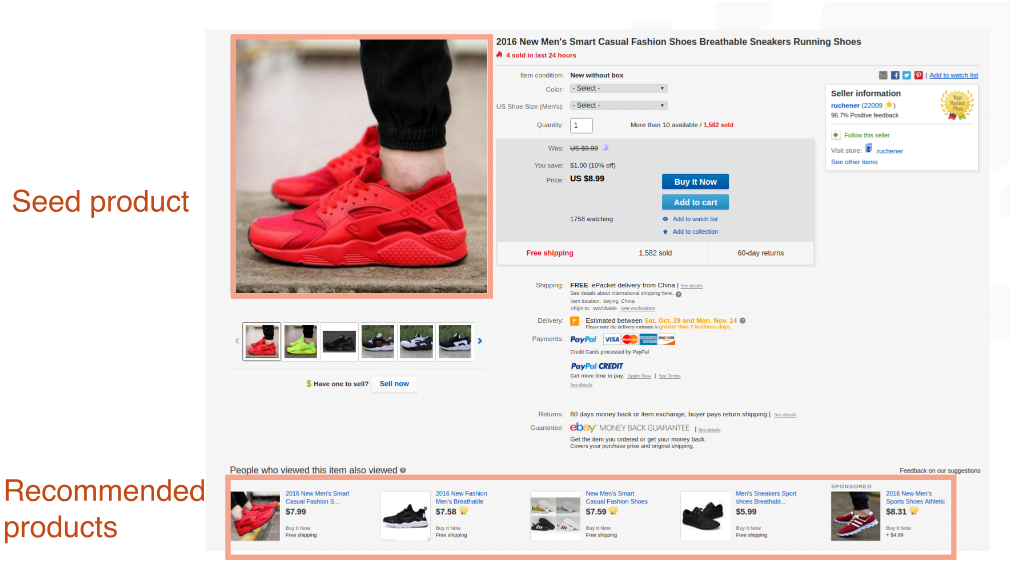
Similarity-based clustering benchmarks. We also compared our approach to two similarity-based clustering benchmarks: (1) k-medoids clustering (de Hoon et al. 2004), an extension of the popular k-means algorithm to handle missing entries, and (2) spectral clustering (Ng et al. 2002) which clusters data points via spectral decomposition of an affinity (or similarity) matrix. We outperform both benchmarks, with upto 84% improvement over k-medoids (for action genre) and 30% over spectral clustering (for comedy genre). Refer to Appendix D.1 for more details.

6. Case study 2: Personalized Recommendations on eBay

In this section, we use our segmentation methodology to *personalize* similar product recommendations on eBay. When a user is on a product page, eBay recommends products that are “similar” to the product being viewed (the *seed* product); see Figure 2 for an example. The recommended products are shown below the seed product, but above the fold[11]. Even without personalization, determining similar products is a challenging task because the product listings offered on the eBay marketplace are diverse. They range from Fitbit tracker/iPhone (products with well-defined attributes) to obscure antiques and collectibles that are highly unstructured, and they can have multiple conditions—new, used, refurbished, etc.—and selling formats—fixed price vs auction. In addition, many products tend to be short-lived—they surface on the site for one week and are never listed again. Nevertheless, Brovman et al. (2016) were able to address these challenges in a scalable recommendation system

[11] Above the fold refers to the portion of the webpage that is visible without scrolling

Figure 2 Example of similar product recommendation on eBay



2016 New Men's Smart Casual Fashion Shoes Breathable Sneakers Running Shoes

4 sold in last 24 hours

Item condition: New without box

Color: - Select -

US Shoe Size (Men's): - Select -

Quantity: 1 More than 10 available / 1,582 sold

Was: ~~US $9.99~~

You save: $1.00 (10% off)

Price: US $8.99

Buy It Now

Add to cart

1758 watching

Add to watch list

Add to collection

Free shipping | 1,582 sold | 60-day returns

Add to watch list

Seller information
ruchener (22009 ★)
96.7% Positive feedback
Follow this seller
Visit store: ruchener
See other items

Shipping: FREE ePacket delivery from China | See details
See details about international shipping here.
Item location: beijing, China
Ships to: Worldwide See exclusions

Delivery: Estimated between Sat. Oct. 29 and Mon. Nov. 14
Please note the delivery estimate is greater than 7 business days.

Payments: PayPal VISA
Credit Cards processed by PayPal

PayPal CREDIT
Get more time to pay. Apply Now | See Terms
See details

$ Have one to sell? Sell now

Returns: 60 days money back or item exchange, buyer pays return shipping | See details

Guarantee: ebay MONEY BACK GUARANTEE | See details
Get the item you ordered or get your money back.
Covers your purchase price and original shipping.

People who viewed this item also viewed

Feedback on our suggestions

2016 New Men's Smart Casual Fashion S...
$7.99
Buy It Now
Free shipping

2016 New Fashion Men's Breathable
$7.58
Buy It Now
Free shipping

New Men's Smart Casual Fashion Shoes
$7.59
Buy It Now
Free shipping

Men's Sneakers Sport shoes Breathabl...
$5.99
Buy It Now
Free shipping

SPONSORED
2016 New Men's Sports Shoes Athletic
$8.31
Buy It Now
+ $4.99

Note. The seed product is a pair of sneakers that the user is currently viewing, and the recommended products at the bottom are other pairs of shoes similar to the seed product.

they implemented at eBay. While their approach resulted in positive lift in critical operational metrics, it does not take into account heterogeneous user preferences—for a given seed product *every* user is recommended the *same* set of products. Personalizing the recommendations to individual users is however challenging because most users interact with only a small fraction of the catalog—in our sample dataset below, a user on average interacted with only 5 out of ~ 4.2M items. Such sparsity makes it hard to determine individual preferences and limits the application of traditional collaborative filtering algorithms. We are able to use our segmentation methodology to address this challenge. We show that segmenting the user population using our technique and personalizing the recommendations to each segment can result in upto 8% improvement in the recommendation quality. It matters how we obtain the segments because other natural approaches to segmentation that are based on similarity in demographics (age, gender, income) and aggregate purchase behavior resulted at best in only $\sim 1\%$ improvement.

Data. The raw data consist of impressions collected over a two-week period in the summer of 2016. An impression is generated whenever a user interacts, either by clicking or clicking and then purchasing, with a recommended product on a product page. The impression contains information on the user, the seed product, the recommended (reco) products, and the actions (clicks and purchases) taken by the user on the reco products. The data provided to us were a random subset of all impressions that had at least one click on a reco product. We transform these data into a preference graph as follows. For each impression, we extract the user and the set of (seed, reco) product pairs. We assign the users to the nodes on the left and the (seed, reco) pairs to the item nodes on the right. We add an edge from a user node to a (seed, reco) node if there is an impression in which the user has taken an action on the (seed, reco) pair, and assign a binary label to the edge: the +1 label if the user clicked and then purchased the reco product and the −1 label if the user did not purchase the reco product, irrespective of whether it was clicked or not. The resulting preference graph consists of ~ 1M users, ~ 4.2M items, i.e., (seed, reco) product pairs, and ~ 4.5M edges across different product categories on eBay. For our analysis, we focus on the two categories with the most numbers of purchases—`Clothing, Shoes and Accessories (CSA)` and `Home & Garden (HG)`. In each category, we randomly split all the binary purchase/no-purchase signals into training (80%) and test (20%) sets. Table 4 reports the summary statistics of the data.

Current approach. Brovman et al. transformed the problem of generating similar recommendations into the following prediction problem: for a fixed seed product, what is the probability that a candidate reco product is purchased? The candidate recommendations are then ranked (in real-time) according to the probability of being purchased. The prediction problem was solved by learning a classifier on the above purchase/no-purchase user interaction data. In particular, they assumed that the population is homogeneous and therefore estimate a single logistic regression classifier[12] on the user interaction data. The authors performed feature engineering to ensure the classifier had good performance; refer to their paper for more details. Let U denote the set of all users and $N^{\text{train}}(i)$ denote the set of items that user $i \in U$ has interacted with in the training data. Let $\boldsymbol{y}_{s,r} \in \mathbb{R}^D$ represent the feature vector corresponding to item (s,r) where s is the seed product and r is the reco product. Further, let $x_{i,(s,r)} \in \{-1,+1\}$ denote the binary purchase (+1) or no-purchase (−1) signal associated with a user i that interacted with item (s,r). Then, logistic regression estimates a parameter vector $\boldsymbol{\omega}_{\text{pool}} \in \mathbb{R}^D$ by solving:

$$\min_{\boldsymbol{\omega}} \sum_{(s,r)} \sum_{i:(s,r)\in N^{\text{train}}(i)} -\mathbf{1}[x_{i,(s,r)} = +1] \cdot \log p_{s,r}(\boldsymbol{\omega}) - \mathbf{1}[x_{i,(s,r)} = -1] \cdot \log\left(1 - p_{s,r}(\boldsymbol{\omega})\right),$$

where $p_{s,r}(\boldsymbol{\omega}) = \frac{\exp(\boldsymbol{\omega}^\top \boldsymbol{y}_{s,r})}{1+\exp(\boldsymbol{\omega}^\top \boldsymbol{y}_{s,r})}$ represents the probability of purchase signal on item (s,r) under parameter $\boldsymbol{\omega}$. We call this approach the *population model*, f_{pool}.

Our approach: capturing heterogeneity through segmentation. The extreme sparsity[13] of the preference graph limits the applicability of most existing clustering techniques. We applied our model-based embedding technique to segment the user population using the preference graph

[12] The authors tried out several different binary classifiers and observed that a logistic regression model achieves comparable performance to more sophisticated methods like random forests and boosted decision trees, and consequently, decided to pick the logistic regression classifier for its simplicity and scalability.

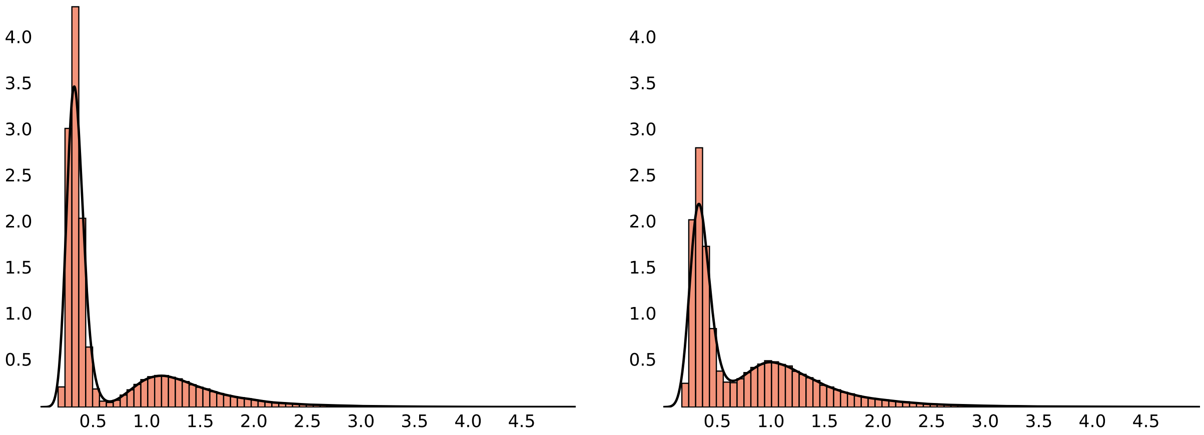
[13] The average sparsity in our dataset is $\sim 0.0007\%$ which is orders of magnitude smaller than in our MovieLens case study as well as the typical sparsity in Netflix-like rating systems (Bell and Koren 2007)

Table 4 AUC improvements of segment-specific logistic regression classifiers over population model

Product Category	Train Data			Test Data		Segments	% AUC imp.	Time to segment
	Users	Items	Edges	Items	Edges			
CSA	172K	534K	561K	116K	118K	Segment 1 (120K) Segment 2 (46K) Segment 3 (6K)	-0.27 0.54 5.78	~6 mins
HG	129K	432K	458K	100K	102K	Segment 1 (78K) Segment 2 (45K) Segment 3 (6K)	0.08 -0.37 7.95	~5 mins

The numbers in parentheses denote the size (in thousands) of each user segment.

Figure 3 Density of user embedding scores for different categories, **Left**: CSA and **Right**: HG



Note. The x-axis, representing the user embedding scores, is cutoff at 5 so that the two modes above can be visualized better. The largest embedding score is 28.3 for the CSA category and 25.6 for the HG category

constructed above. The pooled model corresponds to the aforementioned logistic regression classifier f_{pool}; then we apply Algorithm 3 to compute the embeddings of each user i:

$$v_i = \frac{\sum_{(s,r)\in N^{\text{train}}(i)} -\mathbf{1}[x_{i,(s,r)} = +1] \cdot \log p_{s,r}(\boldsymbol{\omega}_{\text{pool}}) - \mathbf{1}[x_{i,(s,r)} = -1] \cdot \log\left(1 - p_{s,r}(\boldsymbol{\omega}_{\text{pool}})\right)}{\sum_{(s,r)\in N^{\text{train}}(i)} H\left(p_{s,r}(\boldsymbol{\omega}_{\text{pool}})\right)}$$

where $H(\cdot)$ is the binary entropy function introduced in Section 3.1. We perform k-means clustering on the embeddings to obtain the different segments. Once we obtain the segments, we estimate a separate parameter $\boldsymbol{\omega}_k$ for each segment k by fitting a logistic regression classifier f_k using only the interactions of users in segment k.

Results and Discussion. Figure 3 shows the distribution of the user embeddings computed by our algorithm. While one can visually see two modes in the density plots, we also observed a long tail and consequently, chose $K = 3$ segments to account for those users. Brovman et al. used the area-under-the-curve (AUC) metric[14] to evaluate the performance of the logistic regression classifier. For each user segment k, we compare the AUC of the population model f_{pool} with that of the segment-specific classifier f_k, on the test set. Table 4 shows the percentage improvement in AUC of the segment-specific classifiers across each category. We observe that our segmentation technique can lead to significant improvements in the AUC—upto 6% in category `CSA` and 8% in category `HG`. The improvements are the largest for segment 3 or the *esoteric* segment, which consists of users having the highest embedding scores. This, in turn, means that they deviate most from the population preferences and therefore the population model is not able to capture the preferences of these users. Segment 1 or the *mainstream* segment, consists of users having the lowest embedding scores; their preferences are captured well by the population model and therefore the benefit from segmentation is negligible[15]. Segment 2 consists of users who have "intermediate" preferences; they agree with the population on some items but deviate on other items. Their preferences are partially captured by the population model and depending on whether they have more mainstream or esoteric preferences, treating such users separately can result in loss (as for `HG` category) or gain (as for `CSA` category) in performance.

The above improvement obtained using our segmentation technique is non-trivial considering the fact that we also tried several natural approaches such as segmentation based on similarities in demographics (age, gender, income, etc.) and aggregate purchase statistics (no. of transactions and/or amount spent in the last year, etc.). But the best of these resulted in around 1% improvement in AUC for any user segment, compared to the population model (refer to Appendix E for more details). Such approaches implicitly assume that similarity in demographics or aggregate purchase behavior implies similarity in preferences, which might not be the case in practice. Instead, focusing on actual user activity such as click and purchase signals can help to directly capture their preferences. However, a major challenge in using such data is that it is extremely sparse; for instance, in the dataset above, users had only 4-5 observations on average and consequently, most of the users do not have any overlap in the observations that they generate. This makes it hard to determine whether two users have similar preferences. Further, existing techniques like the LC method are prohibitively slow for such a large dataset. Table 4 also shows the time taken to segment the population using our technique, without any optimizations or parallel processing. We believe our implementation can be easily ported to large-scale distributed data processing frameworks like Apache Spark to obtain further speedups. This shows that our segmentation technique can scale to large datasets and work directly with fine-grained user observations (such as click and purchase signals) to generate personalized product recommendations.

[14] AUC measures classifier performance as equal to the likelihood that the classifier will rank (based on the probability of positive class label) a randomly chosen positive example higher than a randomly chosen negative example

[15] For `CSA`, the AUC is (slightly) lower and we attribute this to having smaller number of samples in the training data for the segment-specific classifier

7. Conclusions

This paper presents a novel method to segment customers based on their preferences. Our method is designed to incorporate observations from diverse data sources such as purchases, ratings, clicks, etc. as well as handle missing observations. We propose a *model-based embedding technique* that relies on a probabilistic model class that describes how the observations are generated, to transform the customer observations into a consistent and comparable scale, and then deals with the missing data issue by projecting the transformed data into a low-dimensional space. The low-dimensional representations, called *embeddings*, are then clustered using any existing technique to obtain the segments. Our technique builds upon existing ideas in the machine learning literature for clustering observations from a mixture model (such as Gaussian mixtures) and extends them to handle categorical data, as well as missing entries in the observations. Our method can also leverage the existing literature in marketing that has proposed rich models to capture detailed and fine-grained customer preference structures. A key feature of our segmentation algorithm is that it is analytically tractable, and we derive precise necessary and sufficient conditions in order to guarantee asymptotic recovery of the true customer segments. Experiments on synthetic data show an improvement in the accuracy of recovering the true segments, over the standard latent class benchmark, in conjunction with an order of magnitude speedup. Further, using two case studies, including a real-world implementation on eBay data, we show that our segmentation approach can be used to generate high-quality personalized recommendations.

There are a few natural directions and opportunities for future work. We focused on categorical data in this paper—since most of the observations collected about customers from firms is categorical—but our methodology can also be applied directly to continuous data, and it will be interesting to explore how our algorithm performs when there is a mix of categorical and continuous observations. From the analytical perspective, it will be interesting to determine other generative models (especially from the exponential family) for customer observations under which our algorithm can recover the true segments. For instance, we could consider mixtures of binary logit models where each item j is represented as a vector $\boldsymbol{y}_j$ in some feature space $\mathcal{Y}$. Imposing suitable constraints on the space $\mathcal{Y}$ as well as defining appropriate missing data mechanisms in the customer observations will be important in this regard. More broadly, the idea of separating customers based on their deviation from the population preference can be applied in other domains such as text reviews, images or even audio/speech, to obtain interesting "domain-specific" notions of mainstream and esoteric segments. Finally, it would be useful to test the effectiveness of our segmentation method in terms of standard marketing performance measures such as customer lifetime value, profitability, loyalty, etc.

Acknowledgments

The results on the eBay dataset were achieved during an internship by Ashwin Venkataraman in collaboration with his mentors Yuri Brovman and Natraj Srinivasan, and the authors would like to thank them and eBay for their help and feedback.

References

Abu-Mostafa, Yaser S., Malik Magdon-Ismail, Hsuan-Tien Lin. 2012. *Learning From Data*. AMLBook.

Achlioptas, Dimitris, Frank McSherry. 2005. On spectral learning of mixtures of distributions. *Learning Theory*. Springer, 458–469.

Anandkumar, Animashree, Rong Ge, Daniel Hsu, Sham M Kakade, Matus Telgarsky. 2014. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research* **15**(1) 2773–2832.

Assael, Henry. 1970. Segmenting markets by group purchasing behavior: an application of the aid technique. *Journal of Marketing Research* 153–158.

Bell, Robert M, Yehuda Koren. 2007. Lessons from the netflix prize challenge. *Acm Sigkdd Explorations Newsletter* **9**(2) 75–79.

Brovman, Yuri M., Marie Jacob, Natraj Srinivasan, Stephen Neola, Daniel Galron, Ryan Snyder, Paul Wang. 2016. Optimizing similar item recommendations in a semi-structured marketplace to maximize conversion. *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys '16, ACM, 199–202.

Comaniciu, Dorin, Peter Meer. 2002. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence* **24**(5) 603–619.

de Hoon, Michiel JL, Seiya Imoto, John Nolan, Satoru Miyano. 2004. Open source clustering software. *Bioinformatics* **20**(9) 1453–1454.

DeSarbo, Wayne S, Ajay K Manrai, Lalita A Manrai. 1994. Latent class multidimensional scaling. a review of recent developments in the marketing and psychometric literature. *Advanced Methods of Marketing Research, R. Bagozzi (Ed.), Blackwell Pub* 190–222.

Dhillon, Inderjit S, Yuqiang Guan, Brian Kulis. 2004. Kernel k-means: spectral clustering and normalized cuts. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 551–556.

Filippone, Maurizio, Francesco Camastra, Francesco Masulli, Stefano Rovetta. 2008. A survey of kernel and spectral methods for clustering. *Pattern recognition* **41**(1) 176–190.

Fraley, Chris, Adrian E Raftery. 2002. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American statistical Association* **97**(458) 611–631.

Gupta, Sachin, Pradeep K Chintagunta. 1994. On using demographic variables to determine segment membership in logit mixture models. *Journal of Marketing Research* 128–136.

Herlocker, Jonathan L, Joseph A Konstan, Al Borchers, John Riedl. 1999. An algorithmic framework for performing collaborative filtering. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 230–237.

Hsu, Daniel, Sham M Kakade. 2013. Learning mixtures of spherical gaussians: moment methods and spectral decompositions. *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*. ACM, 11–20.

Jain, Anil K. 2010. Data clustering: 50 years beyond k-means. *Pattern recognition letters* **31**(8) 651–666.

Kamakura, Wagner A. 1988. A least squares procedure for benefit segmentation with conjoint experiments. *Journal of Marketing Research* **25** 157–67.

Kamakura, Wagner A, Byung-Do Kim, Jonathan Lee. 1996. Modeling preference and structural heterogeneity in consumer choice. *Marketing Science* **15**(2) 152–172.

Kamakura, Wagner A, Gary Russell. 1989. A probabilistic choice model for market segmentation and elasticity structure. *Journal of Marketing Research* **26** 379–390.

Kannan, Ravindran, Hadi Salmasian, Santosh Vempala. 2005. The spectral method for general mixture models. *Learning Theory*. Springer, 444–457.

Koren, Yehuda, Robert Bell, Chris Volinsky, et al. 2009. Matrix factorization techniques for recommender systems. *Computer* **42**(8) 30–37.

Lin, Jovian, Kazunari Sugiyama, Min-Yen Kan, Tat-Seng Chua. 2013. Addressing cold-start in app recommendation: latent user models constructed from twitter followers. *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 283–292.

Maclachlan, Douglas L, Johny K Johansson. 1981. Market segmentation with multivariate aid. *The Journal of Marketing* 74–84.

Mazumder, Rahul, Trevor Hastie, Robert Tibshirani. 2010. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research* **11**(Aug) 2287–2322.

McLachlan, Geoffrey, David Peel. 2004. *Finite mixture models*. John Wiley & Sons.

Ng, Andrew Y, et al. 2002. On spectral clustering: Analysis and an algorithm .

Ogawa, Kohsuke. 1987. An approach to simultaneous estimation and segmentation in conjoint analysis. *Marketing Science* **6**(1) 66–81.

Park, Seung-Taek, Wei Chu. 2009. Pairwise preference regression for cold-start recommendation. *Proceedings of the third ACM conference on Recommender systems*. ACM, 21–28.

Rokach, Lior, Oded Maimon. 2005. Clustering methods. *Data mining and knowledge discovery handbook*. Springer, 321–352.

Rossi, Peter E, Greg M Allenby, Rob McCulloch. 2005. *Bayesian statistics and marketing*. John Wiley & Sons.

Schein, Andrew I, Alexandrin Popescul, Lyle H Ungar, David M Pennock. 2002. Methods and metrics for cold-start recommendations. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 253–260.

Sedhain, Suvash, Scott Sanner, Darius Braziunas, Lexing Xie, Jordan Christensen. 2014. Social collaborative filtering for cold-start recommendations. *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 345–348.

Shani, Guy, Asela Gunawardana. 2011. Evaluating recommendation systems. *Recommender systems handbook*. Springer, 257–297.

Shi, Jianbo, Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* **22**(8) 888–905.

Smith, Wendell R. 1956. Product differentiation and market segmentation as alternative marketing strategies. *Journal of marketing* **21**(1) 3–8.

Strehl, Alexander, Joydeep Ghosh. 2002. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research* **3**(Dec) 583–617.

Takács, Gábor, István Pilászy, Bottyán Németh, Domonkos Tikk. 2009. Scalable collaborative filtering approaches for large recommender systems. *Journal of machine learning research* **10**(Mar) 623–656.

Von Luxburg, Ulrike. 2007. A tutorial on spectral clustering. *Statistics and computing* **17**(4) 395–416.

Wainwright, Martin J, Michael I Jordan, et al. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning* **1**(1–2) 1–305.

Wang, Junhui. 2010. Consistent selection of the number of clusters via crossvalidation. *Biometrika* **97**(4) 893–904.

Wedel, Michel, Wayne S DeSarbo. 1994. A review of recent developments in latent class regression models. *Advanced methods of marketing research* 352–388.

Wedel, Michel, Wagner A Kamakura. 2000. *Market segmentation: Conceptual and methodological foundations*, vol. 8. Springer Science & Business Media.

Wedel, Michel, Cor Kistemaker. 1989. Consumer benefit segmentation using clusterwise linear regression. *International Journal of Research in Marketing* **6**(1) 45–59.

Wedel, Michel, Jan-Benedict EM Steenkamp. 1989. A fuzzy clusterwise regression approach to benefit segmentation. *International Journal of Research in Marketing* **6**(4) 241–258.

Wright, John, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S Huang, Shuicheng Yan. 2010. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE* **98**(6) 1031–1044.

Wu, Sen, Xiaodong Feng, Wenjun Zhou. 2014. Spectral clustering of high-dimensional data exploiting sparse representation vectors. *Neurocomputing* **135** 229–239.

Xu, Rui, Donald Wunsch. 2005. Survey of clustering algorithms. *IEEE Transactions on neural networks* **16**(3) 645–678.

Zhang, Mi, Jie Tang, Xuchen Zhang, Xiangyang Xue. 2014. Addressing cold start in recommender systems: A semi-supervised co-training algorithm. *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 73–82.

Zhong, Shi, Joydeep Ghosh. 2003. A unified framework for model-based clustering. *The Journal of Machine Learning Research* **4** 1001–1037.

A Model-based Embedding Technique for Segmenting Customers

Appendix

Srikanth Jagabathula
Leonard N. Stern School of Business, New York University,
44 West Fourth St., New York, NY 10012
sjagabat@stern.nyu.edu
Lakshmi Subramanian Ashwin Venkataraman
Courant Institute of Mathematical Sciences, New York University,
251 Mercer Street, New York, NY 10012
{lakshmi,ashwin}@cs.nyu.edu

We begin by proving some general statements about random variables, that will be used in the proofs later.

Lemma A1. *Let $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_r, \boldsymbol{Y}$ be a collection of non-negative random variables. Let $a_1, a_2, \ldots, a_r, b$ be positive constants. Then, given any $0 < \varepsilon < 1$ and any $1 \le i \le r$, we have that*

$$(i)\, \Pr\left[\left|\frac{\boldsymbol{X}_i}{\boldsymbol{Y}} - \frac{a_i}{b}\right| > \varepsilon \frac{a_i}{b}\right] \le \Pr\left[|\boldsymbol{X}_i - a_i| > \varepsilon' a_i\right] + \Pr\left[|\boldsymbol{Y} - b| > \varepsilon' b\right]$$

$$(ii)\, \Pr\left[|\boldsymbol{X}_i \boldsymbol{Y} - a_i b| > \varepsilon a_i b\right] \le \Pr\left[|\boldsymbol{X}_i - a_i| > \varepsilon' a_i\right] + \Pr\left[|\boldsymbol{Y} - b| > \varepsilon' b\right]$$

$$(iii)\, \Pr\left[\left|\sum_{i=1}^r \boldsymbol{X}_i - \sum_{i=1}^r a_i\right| > \varepsilon \cdot \left(\sum_{i=1}^r a_i\right)\right] \le \sum_{i=1}^r \Pr\left[|\boldsymbol{X}_i - a_i| > \varepsilon a_i\right]$$

$$(iv)\, \Pr\left[\sum_{i=1}^r |\boldsymbol{X}_i - a_i| > \varepsilon \cdot \left(\sum_{i=1}^r a_i\right)\right] \le \sum_{i=1}^r \Pr\left[|\boldsymbol{X}_i - a_i| > \varepsilon a_i\right]$$

where $\varepsilon' = \varepsilon/3$.

Proof. **Part (i)**. Let $\boldsymbol{Z}_1 = \frac{\boldsymbol{X}_1}{\boldsymbol{Y}}$. We prove the result by contradiction. Suppose $\boldsymbol{Z}_1 > (1+\varepsilon)\frac{a_1}{b}$. Then, $\boldsymbol{X}_1 > (1+\varepsilon')a_1$ or $\boldsymbol{Y} < (1-\varepsilon')b$. If not, we have the following:

$$\begin{aligned} \boldsymbol{X}_1 \le (1+\varepsilon')a_1,\ \boldsymbol{Y} \ge (1-\varepsilon')b &\implies \frac{\boldsymbol{X}_1}{\boldsymbol{Y}} \le \frac{(1+\varepsilon')a_1}{(1-\varepsilon')b} \\ &\implies \boldsymbol{Z}_1 \le (1+\varepsilon)\frac{a_1}{b} \end{aligned}$$

where the last implication follows from the fact that $\frac{1+\varepsilon'}{1-\varepsilon'} \le 1+\varepsilon$ when $\varepsilon' = \varepsilon/3$ and $0 < \varepsilon < 1$. This is a contradiction. Therefore we have that,

$$\Pr\left[\boldsymbol{Z}_1 > (1+\varepsilon)\frac{a_1}{b}\right] \le \Pr\left[\boldsymbol{X}_1 > (1+\varepsilon')a_1 \bigcup \boldsymbol{Y} < (1-\varepsilon')b\right] \le \Pr\left[\boldsymbol{X}_1 > (1+\varepsilon')a_1\right] + \Pr\left[\boldsymbol{Y} < (1-\varepsilon')b\right]$$

where the last inequality follows from the union bound. An analogous argument establishes that

$$\Pr\left[\boldsymbol{Z}_1 < (1-\varepsilon)\frac{a_1}{b}\right] \leq \left(\Pr\left[\boldsymbol{X}_1 < (1-\varepsilon')a_1\right] + \Pr\left[\boldsymbol{Y} > (1+\varepsilon')b\right]\right)$$

which uses the fact that $\frac{1-\varepsilon'}{1+\varepsilon'} \geq 1-\varepsilon$ when $\varepsilon' = \varepsilon/3$ and $0 < \varepsilon < 1$. Combining the above two arguments, we get

$$\Pr\left[\left|\boldsymbol{Z}_1 - \frac{a_1}{b}\right| > \varepsilon\frac{a_1}{b}\right] \leq \left(\Pr\left[|\boldsymbol{X}_1 - a_1| > \varepsilon' a_1\right] + \Pr\left[|\boldsymbol{Y} - b| > \varepsilon' b\right]\right)$$

Part (ii). Let $\boldsymbol{W}_1 = \boldsymbol{X}_1\boldsymbol{Y}$. Suppose $\boldsymbol{W}_1 > (1+\varepsilon)a_1 b$. Then, $\boldsymbol{X} > (1+\varepsilon')a_1$ or $\boldsymbol{Y} > (1+\varepsilon')b$. If not, we have the following:

$$\begin{aligned}\boldsymbol{X} \leq (1+\varepsilon')a_1,\ \boldsymbol{Y} \leq (1+\varepsilon')b &\implies \boldsymbol{X}_1\boldsymbol{Y} \leq (1+\varepsilon')^2 a_1 b \\ &\implies \boldsymbol{W}_1 \leq (1+\varepsilon)a_1 b\end{aligned}$$

where the last implication follows because $(1+\varepsilon')^2 \leq 1+\varepsilon$ for $\varepsilon' = \varepsilon/3$ and $0 < \varepsilon < 1$. This is a contradiction. Therefore,

$$\Pr\left[\boldsymbol{W}_1 > (1+\varepsilon)a_1 b\right] \leq \Pr\left[\boldsymbol{X}_1 > (1+\varepsilon')a_1 \bigcup \boldsymbol{Y} > (1+\varepsilon')b\right] \leq \Pr\left[\boldsymbol{X}_1 > (1+\varepsilon')a_1\right] + \Pr\left[\boldsymbol{Y} > (1+\varepsilon')b\right]$$

Combining with the symmetric case gives:

$$\Pr\left[|\boldsymbol{W}_1 - a_1 b| > \varepsilon a_1 b\right] \leq \left(\Pr\left[|\boldsymbol{X}_1 - a_1| > \varepsilon' a_1\right] + \Pr\left[|\boldsymbol{Y} - b| > \varepsilon' b\right]\right)$$

Part (iii). Define $\boldsymbol{Z} \overset{\text{def}}{=} \sum_{i=1}^r \boldsymbol{X}_i$ and $A \overset{\text{def}}{=} \sum_{i=1}^r a_i$. Suppose that $\boldsymbol{Z} > (1+\varepsilon)A$. Then it follows that for some $1 \leq i \leq r$, $\boldsymbol{X}_i > (1+\varepsilon)a_i$. If not, we have:

$$\begin{aligned}\boldsymbol{X}_1 \leq (1+\varepsilon)a_1, \ldots \boldsymbol{X}_r \leq (1+\varepsilon)a_r &\implies \sum_{i=1}^r \boldsymbol{X}_i \leq \sum_{i=1}^r (1+\varepsilon)a_i \\ &\implies \boldsymbol{Z} \leq (1+\varepsilon)A\end{aligned}$$

which is a contradiction. Combining with the symmetric case and applying the union bound, the claim follows.

Part (iv). Let $\boldsymbol{Z} \overset{\text{def}}{=} \sum_{i=1}^r |\boldsymbol{X}_i - a_i|$ and suppose $\boldsymbol{Z} > \varepsilon \cdot (\sum_{i=1}^r a_i)$. Then it follows that for some $1 \leq i \leq r$, $|\boldsymbol{X}_i - a_i| > \varepsilon a_i$. If not, we have:

$$\begin{aligned}|\boldsymbol{X}_1 - a_1| \leq \varepsilon a_1, \ldots |\boldsymbol{X}_r - a_r| \leq \varepsilon a_r &\implies \sum_{i=1}^r |\boldsymbol{X}_i - a_i| \leq \sum_{i=1}^r \varepsilon a_i \\ &\implies \boldsymbol{Z} \leq \varepsilon \cdot \left(\sum_{i=1}^r a_i\right)\end{aligned}$$

which is a contradiction. The claim then follows from the union bound. □

Lemma A2. *Let $\boldsymbol{X}, \boldsymbol{Y}$ be two non-negative random variables such that $0 \le \boldsymbol{X}, \boldsymbol{Y} \le 1$ and $\boldsymbol{Y} = 0 \implies \boldsymbol{X} = 0$. Suppose that $\mathbb{E}[\boldsymbol{X}], \mathbb{E}[\boldsymbol{Y}] > 0$. Define the random variables $\boldsymbol{Z}_1 = \boldsymbol{X} \cdot (-\log \boldsymbol{Y})$ and $\boldsymbol{Z}_2 = \boldsymbol{X} \cdot (-\log \boldsymbol{X})$, and the constants $A_1 = \mathbb{E}[\boldsymbol{X}] \cdot (-\log \mathbb{E}[\boldsymbol{Y}])$ and $A_2 = \mathbb{E}[\boldsymbol{X}] \cdot (-\log \mathbb{E}[\boldsymbol{X}])$. Then, given any $0 < \varepsilon < 1$, we have:*

$$\Pr\left[|\boldsymbol{Z}_1 - A_1| > \varepsilon A_1\right] \le \Pr\left[|\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}]| > \frac{\varepsilon}{3}\mathbb{E}[\boldsymbol{X}]\right] + \Pr\left[|(-\log \boldsymbol{Y} - (-\log \mathbb{E}[\boldsymbol{Y}])| > \frac{\varepsilon}{3} \cdot (-\log \mathbb{E}[\boldsymbol{Y}])\right]$$

$$\Pr\left[|\boldsymbol{Z}_2 - A_2| > \varepsilon A_2\right] \le \Pr\left[|\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}]| > \frac{\varepsilon}{3}\mathbb{E}[\boldsymbol{X}]\right] + \Pr\left[|(-\log \boldsymbol{X} - (-\log \mathbb{E}[\boldsymbol{X}])| > \frac{\varepsilon}{3} \cdot (-\log \mathbb{E}[\boldsymbol{X}])\right]$$

Proof. Note that since $\boldsymbol{X}, \boldsymbol{Y} \in [0,1]$, it follows that $-\log \boldsymbol{X}, -\log \boldsymbol{Y}$ are non-negative. Also, since $\boldsymbol{Y} = 0 \implies \boldsymbol{X} = 0$, the random variables $\boldsymbol{Z}_1, \boldsymbol{Z}_2$ are both well-defined (with the convention that $x \log x = 0$ when $x = 0$). Further, since we also have $0 < \mathbb{E}[\boldsymbol{X}], \mathbb{E}[\boldsymbol{Y}] < 1$, so that $-\log \mathbb{E}[\boldsymbol{X}] > 0$ and $-\log \mathbb{E}[\boldsymbol{Y}] > 0$. The claims then follow from a straightforward application of part (ii) of Lemma A1.

□

Lemma A3. *Let $0 \le \boldsymbol{X} \le 1$ be a non-negative random variable with $1 > \mathbb{E}[\boldsymbol{X}] > 0$. Then given any $0 < \varepsilon < 1$, for all values of the random variable $\boldsymbol{X}$ in the interval $I := \Big((1-\varepsilon')\mathbb{E}[\boldsymbol{X}], (1+\varepsilon')\mathbb{E}[\boldsymbol{X}]\Big)$, where $\varepsilon' = \frac{-\varepsilon \log \mathbb{E}[\boldsymbol{X}]}{1 - \varepsilon \log \mathbb{E}[\boldsymbol{X}]}$, we have*

$$|-\log(\boldsymbol{X}) - (-\log \mathbb{E}[\boldsymbol{X}])| \le \varepsilon \cdot (-\log \mathbb{E}[\boldsymbol{X}])$$

Proof. Since $0 < \mathbb{E}[\boldsymbol{X}] < 1$, it means that $-\log \mathbb{E}[\boldsymbol{X}] > 0$ and consequently, $0 < \varepsilon' < 1$. In addition, it can be seen that $(1+\varepsilon')\mathbb{E}[\boldsymbol{X}] \le 1$ for any $0 < \varepsilon < 1$ and any $0 < \mathbb{E}[\boldsymbol{X}] < 1$, so that $I \subset [0,1]$. Consider the function $g(x) = -\log x$ and note that it is continuous and differentiable on the interval I. The Mean Value theorem says that given a differentiable function $g(\cdot)$ in the interval (a,b), there exists $c \in (a,b)$ such that

$$\frac{g(b) - g(a)}{b - a} = g'(c) = \frac{g(a) - g(b)}{a - b}$$

where $g'(\cdot)$ is the derivative of $g(\cdot)$. Using the mean value theorem for $g(x) = -\log x$ in the interval I, it follows that for all values of random variable $\boldsymbol{X} \in I$ there exists some $\boldsymbol{Z}$ between $\mathbb{E}[\boldsymbol{X}]$ and $\boldsymbol{X}$ such that

$$\frac{-\log \boldsymbol{X} - (-\log \mathbb{E}[\boldsymbol{X}])}{\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}]} = \frac{-1}{\boldsymbol{Z}}$$

Now since $\boldsymbol{Z} \in I$, it follows that $\frac{1}{\boldsymbol{Z}} \le \frac{1}{(1-\varepsilon')\mathbb{E}[\boldsymbol{X}]}$. Also, since $\boldsymbol{X} \in I$, we have $|\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}]| \le \varepsilon' \mathbb{E}[\boldsymbol{X}]$. Then it follows:

$$|-\log \boldsymbol{X} - (-\log \mathbb{E}[\boldsymbol{X}])| = \left|\frac{-(\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}])}{\boldsymbol{Z}}\right| = \frac{|\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}]|}{\boldsymbol{Z}} \le \frac{|\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}]|}{(1-\varepsilon')\mathbb{E}[\boldsymbol{X}]} \le \frac{\varepsilon'}{1-\varepsilon'} = \varepsilon \cdot (-\log \mathbb{E}[\boldsymbol{X}])$$

□

Appendix A: Latent Class Independent (LC-IND) model

First, we introduce some additional notation. Let $\mathbf{1}[A]$ denote the indicator variable taking value 1 if an event A is true and 0 otherwise. Let $\boldsymbol{X}_i^+$ (resp. $\boldsymbol{X}_i^-$) denote the number of items rated as +1 (resp. −1) by customer i. In other words, $\boldsymbol{X}_i^+ = \sum_{j\in N(i)} \mathbf{1}[\boldsymbol{X}_{ij} = +1]$ and $\boldsymbol{X}_i^- = \sum_{j\in N(i)} \mathbf{1}[\boldsymbol{X}_{ij} = -1]$, where recall that $N(i)$ denotes the set of items rated by customer i. Here, $\boldsymbol{X}_{ij}$ represents the rating provided by customer i for item j, note that it is a random variable under the LC-IND model. Next, let $\boldsymbol{F}_0^+ = \frac{\sum_{i'=1}^m \boldsymbol{X}_{i'}^+}{m\cdot\ell}$, so $\boldsymbol{F}_0^+$ is the fraction of likes (+1s) received from the customer population. Finally, let $\text{Bin}(r,p)$ denote the Binomial distribution with parameters r and p.

We begin by proving a result that will be used in the proof later.

Lemma A4. *Consider the random variable* $\boldsymbol{F}_0^+ = \frac{\sum_{i'=1}^m \boldsymbol{X}_{i'}^+}{m\cdot\ell}$. *Given any* $t > 0$, *the following facts are true:*

$$(i)\ \mathbb{E}[\boldsymbol{F}_0^+] = \alpha_{\text{pool}} \qquad (ii)\ \Pr\left[\left|\boldsymbol{F}_0^+ - \alpha_{\text{pool}}\right| \geq t\right] \leq 2\exp\left(-2m\ell t^2\right)$$

Proof. We begin with the expectation:

$$\mathbb{E}[\boldsymbol{F}_0^+] = \frac{\sum_{i'=1}^m \mathbb{E}[\boldsymbol{X}_{i'}^+]}{m\cdot\ell} = \frac{\sum_{i'=1}^m \ell\alpha_{z_i'}}{m\cdot\ell} = \frac{\sum_{k'=1}^K q_{k'} m\cdot(\ell\alpha_{k'})}{m\cdot\ell} = \sum_{k'=1}^K q_{k'}\alpha_{k'} = \alpha_{\text{pool}}$$

using the fact that proportion $q_{k'}$ of the customer population belongs to segment k'. For part (ii), observe that $\boldsymbol{F}_0^+$ can be equivalently written as:

$$\boldsymbol{F}_0^+ = \frac{\sum\limits_{i'=1}^m \sum\limits_{j\in N(i')} \mathbf{1}[\boldsymbol{X}_{ij} = +1]}{m\cdot\ell}$$

In other words, $\boldsymbol{F}_0^+$ is an average of $m\cdot\ell$ random variables, which are independent under the LC-IND model (since each customer rates items independently). Then using Hoeffding's inequality we can show, for any $t > 0$:

$$\Pr\left[\left|\boldsymbol{F}_0^+ - \alpha_{\text{pool}}\right| \geq t\right] \leq 2\exp\left(-2m\ell t^2\right)$$

□

A.1. Concentration of customer embedding scores

Proof of Lemma 1. To calculate the customer embedding scores, we first need to compute the pooled estimate. Since the underlying LC-IND model is parameterized by a single parameter that specifies the probability of liking any item, the pooled estimate is given by $\frac{\sum_{i'=1}^m \boldsymbol{X}_{i'}^+}{m\cdot\ell} = \boldsymbol{F}_0^+$ based on our definition earlier. Also, let us denote the fraction of likes given by customer i as $\boldsymbol{F}_i^+ \overset{\text{def}}{=} \frac{\boldsymbol{X}_i^+}{\ell}$. Then, the uni-dimensional embedding $\boldsymbol{V}_i$ for customer i is given by:

$$\begin{aligned}
\boldsymbol{V}_i &= \frac{-\sum_{j\in N(i)} \big(\mathbf{1}[\boldsymbol{X}_{ij}=+1]\log \boldsymbol{F}_0^+ + \mathbf{1}[\boldsymbol{X}_{ij}=-1]\log(1-\boldsymbol{F}_0^+)\big)}{-\sum_{j\in N(i)} \big(\boldsymbol{F}_0^+\log \boldsymbol{F}_0^+ + (1-\boldsymbol{F}_0^+)\log(1-\boldsymbol{F}_0^+)\big)} \\
&= \frac{-\big(\sum_{j\in N(i)} \mathbf{1}[\boldsymbol{X}_{ij}=+1]\big)\log \boldsymbol{F}_0^+ - \big(\sum_{j\in N(i)} \mathbf{1}[\boldsymbol{X}_{ij}=-1]\big)\log(1-\boldsymbol{F}_0^+)}{-\big(\boldsymbol{F}_0^+\log \boldsymbol{F}_0^+ + (1-\boldsymbol{F}_0^+)\log(1-\boldsymbol{F}_0^+)\big)\cdot \ell} \\
&= \frac{-\big(\frac{\boldsymbol{X}_i^+}{\ell}\big)\log \boldsymbol{F}_0^+ - \big(1-\frac{\boldsymbol{X}_i^+}{\ell}\big)\log(1-\boldsymbol{F}_0^+)}{-\big(\boldsymbol{F}_0^+\log \boldsymbol{F}_0^+ + (1-\boldsymbol{F}_0^+)\log(1-\boldsymbol{F}_0^+)\big)} \\
&= \frac{-\boldsymbol{F}_i^+\log \boldsymbol{F}_0^+ - \big(1-\boldsymbol{F}_i^+\big)\log(1-\boldsymbol{F}_0^+)}{-\big(\boldsymbol{F}_0^+\log \boldsymbol{F}_0^+ + (1-\boldsymbol{F}_0^+)\log(1-\boldsymbol{F}_0^+)\big)}
\end{aligned}$$

Note that when $\boldsymbol{F}_0^+ \in \{0,1\}$, we define $\boldsymbol{V}_i = 0$ which is the limiting value as $\boldsymbol{F}_0^+ \to 0$ or $\boldsymbol{F}_0^+ \to 1$.

Concentration of $\boldsymbol{F}_i^+$. From the generative model, it follows that the random variable representing the number of likes given by customer i is a binomial random variable, i.e. $\boldsymbol{X}_i^+ \sim \text{Bin}(\ell, \alpha_{z_i})$. Then, using Hoeffding's inequality we can show that for any $t > 0$:

$$\begin{aligned}
\Pr\Big[\big|\boldsymbol{F}_i^+ - \alpha_{z_i}\big| \geq t\Big] &\leq 2\exp\big(-2\ell t^2\big) \\
\Pr\Big[\big|\big(1-\boldsymbol{F}_i^+\big) - (1-\alpha_{z_i})\big| \geq t\Big] &\leq 2\exp\big(-2\ell t^2\big)
\end{aligned} \tag{A1}$$

Concentration of $-\log \boldsymbol{F}_0^+$ and $-\log(1-\boldsymbol{F}_0^+)$. Lemma A4 says that $\mathbb{E}[\boldsymbol{F}_0^+] = \alpha_{\text{pool}} \geq \alpha_{\min} > 0$ and observe that $0 \leq \boldsymbol{F}_0^+ \leq 1$. So we can apply Lemma A3 to the random variable $\boldsymbol{F}_0^+$, which says that given any $0 < \varepsilon < 1$, for all values of random variable $\boldsymbol{F}_0^+$ in the interval $\Big(\alpha_{\text{pool}}\cdot(1-\varepsilon'), \alpha_{\text{pool}}\cdot(1+\varepsilon')\Big)$ with $\varepsilon' = \frac{-\varepsilon\log\alpha_{\text{pool}}}{1-\varepsilon\log\alpha_{\text{pool}}}$, we have:

$$\big|-\log \boldsymbol{F}_0^+ - (-\log\alpha_{\text{pool}})\big| \leq \varepsilon\cdot(-\log\alpha_{\text{pool}}) \tag{A2}$$

Now, for any $0 < \varepsilon < 1$, define $t(\varepsilon) \overset{\text{def}}{=} \varepsilon \cdot \left(\frac{-\bar{\alpha}_{\text{pool}}\cdot\log\big(1-\bar{\alpha}_{\text{pool}}\big)}{1-\log\big(1-\bar{\alpha}_{\text{pool}}\big)}\right)$, where recall that $\bar{\alpha}_{\text{pool}} = \min\{\alpha_{\text{pool}}, 1-\alpha_{\text{pool}}\}$, as defined in the statement of the Lemma 1. Note that $\bar{\alpha}_{\text{pool}} < (1/2)$ since $\alpha_{\text{pool}} \neq (1/2)$ and therefore $t(\varepsilon)$ is well-defined. It is easy to check that $0 < t(\varepsilon) \leq \alpha_{\text{pool}}\cdot\varepsilon'$ where note from above that $\varepsilon' = \frac{-\varepsilon\log\alpha_{\text{pool}}}{1-\varepsilon\log\alpha_{\text{pool}}}$. Then using Lemma A4, we get that

$$\Pr\Big[\big|\boldsymbol{F}_0^+ - \alpha_{\text{pool}}\big| \leq \varepsilon'\alpha_{\text{pool}}\Big] \geq \Pr\Big[\big|\boldsymbol{F}_0^+ - \alpha_{\text{pool}}\big| \leq t(\varepsilon)\Big] \geq 1 - 2\exp\big(-2m\ell\cdot t^2(\varepsilon)\big)$$

Then, using equation (A2) it follows that

$$\Pr\Big[\big|-\log \boldsymbol{F}_0^+ - (-\log\alpha_{\text{pool}})\big| \leq \varepsilon\cdot(-\log\alpha_{\text{pool}})\Big] \geq \Pr\Big[\big|\boldsymbol{F}_0^+ - \alpha_{\text{pool}}\big| \leq \varepsilon'\alpha_{\text{pool}}\Big] \geq 1-2\exp\big(-2m\ell\cdot t^2(\varepsilon)\big) \tag{A3}$$

A similar sequence of arguments (using the random variable $1-\boldsymbol{F}_0^+$) shows that for $\varepsilon'' = \frac{-\varepsilon\log(1-\alpha_{\text{pool}})}{1-\varepsilon\log(1-\alpha_{\text{pool}})}$ and observing that $t(\varepsilon) \le (1-\alpha_{\text{pool}})\cdot\varepsilon''$:

$$\begin{aligned}\Pr\left[\left|-\log(1-\boldsymbol{F}_0^+) - (-\log(1-\alpha_{\text{pool}}))\right| \le \varepsilon\cdot(-\log(1-\alpha_{\text{pool}}))\right] &\ge \Pr\left[\left|\boldsymbol{F}_0^+ - \alpha_{\text{pool}}\right| \le \varepsilon''\cdot(1-\alpha_{\text{pool}})\right]\\ &\ge 1-2\exp\big(-2m\ell\cdot t^2(\varepsilon)\big)\end{aligned} \tag{A4}$$

For ease of notation in the remainder of the proof, denote the embedding score $\boldsymbol{V}_i = \frac{\boldsymbol{N}_i}{\boldsymbol{D}_i}$ to specify the numerator and denominator terms.

Concentration of $\boldsymbol{N}_i$. Let us begin with the numerator, $\boldsymbol{N}_i = -\boldsymbol{F}_i^+\log\boldsymbol{F}_0^+ - (1-\boldsymbol{F}_i^+)\log(1-\boldsymbol{F}_0^+)$. Consider the first term: $\boldsymbol{F}_i^+\cdot(-\log\boldsymbol{F}_0^+)$ and note that $\mathbb{E}[\boldsymbol{F}_i^+] = \alpha_{z_i}$, $\mathbb{E}[\boldsymbol{F}_0^+] = \alpha_{\text{pool}}$. Then using Lemma A2 with $\boldsymbol{X} = \boldsymbol{F}_i^+, \boldsymbol{Y} = \boldsymbol{F}_0^+$ and denoting $A_1 = \hat{c}_1 \overset{\text{def}}{=} \alpha_{z_i}\cdot(-\log\alpha_{\text{pool}})$:

$$\begin{aligned}&\Pr\left[\left|\boldsymbol{F}_i^+\cdot(-\log\boldsymbol{F}_0^+) - \hat{c_1}\right| > \varepsilon\hat{c_1}\right]\\ &\le \Pr\left[\left|\boldsymbol{F}_i^+ - \alpha_{z_i}\right| > \frac{\varepsilon}{3}\alpha_{z_i}\right] + \Pr\left[\left|(-\log\boldsymbol{F}_0^+ - (-\log\alpha_{\text{pool}})\right| > \frac{\varepsilon}{3}\cdot(-\log\alpha_{\text{pool}})\right]\\ &\le 2\exp\left(-2\ell\frac{\varepsilon^2\alpha_{z_i}^2}{9}\right) + 2\exp\left(-2m\ell\cdot t^2(\varepsilon/3)\right) \text{ (using equations (A1) and (A3))}\\ &\le 2\exp\left(-2\ell\frac{\varepsilon^2\alpha_{\min}^2}{9}\right) + 2\exp\left(-2m\ell\cdot t^2(\varepsilon/3)\right) \text{ (since } \alpha_{z_i}\ge\alpha_{\min})\end{aligned} \tag{A5}$$

Similarly for the second term, observe that $\mathbb{E}[1-\boldsymbol{F}_i^+] = 1-\alpha_{z_i}$, $\mathbb{E}[1-\boldsymbol{F}_0^+] = 1-\alpha_{\text{pool}}$. Therefore, choosing $\boldsymbol{X} = (1-\boldsymbol{F}_i^+), \boldsymbol{Y} = 1-\boldsymbol{F}_0^+$ and denoting $A_2 = \hat{c}_2 \overset{\text{def}}{=} (1-\alpha_{z_i})\cdot\Big(-\log(1-\alpha_{\text{pool}})\Big)$ in Lemma A2:

$$\begin{aligned}&\Pr\left[\left|(1-\boldsymbol{F}_i^+)\cdot\Big(-\log(1-\boldsymbol{F}_0^+)\Big) - \hat{c_2}\right| > \varepsilon\hat{c_2}\right]\\ &\le \Pr\left[\left|(1-\boldsymbol{F}_i^+) - (1-\alpha_{z_i})\right| > \frac{\varepsilon}{3}\cdot(1-\alpha_{z_i})\right] + \Pr\left[\left|-\log(1-\boldsymbol{F}_0^+) - (-\log(1-\alpha_{\text{pool}}))\right| > \frac{\varepsilon}{3}\cdot(-\log(1-\alpha_{\text{pool}}))\right]\\ &\le 2\exp\left(-2\ell\frac{\varepsilon^2(1-\alpha_{z_i})^2}{9}\right) + 2\exp\left(-2m\cdot t^2(\varepsilon/3)\right) \text{ (using equations (A1) and (A4))}\\ &\le 2\exp\left(-2\ell\frac{\varepsilon^2\alpha_{\min}^2}{9}\right) + 2\exp\left(-2m\cdot t^2(\varepsilon/3)\right) \text{ (since } (1-\alpha_{z_i})\ge\alpha_{\min})\end{aligned} \tag{A6}$$

Combining the above two, choosing $\boldsymbol{X}_1 = \boldsymbol{F}_i^+\cdot(-\log\boldsymbol{F}_0^+)$, $\boldsymbol{X}_2 = (1-\boldsymbol{F}_i^+)\cdot(-\log(1-\boldsymbol{F}_0^+))$, $a_1 = \hat{c}_1$ and $a_2 = \hat{c}_2$ in Lemma A1, we get:

$$\begin{aligned}&\Pr\left[\left|\boldsymbol{N}_i - (\hat{c_1}+\hat{c_2})\right| > \frac{\varepsilon}{3}\cdot(\hat{c_1}+\hat{c_2})\right]\\ &\le \Pr\left[\left|\boldsymbol{F}_i^+\cdot(-\log\boldsymbol{F}_0^+) - \hat{c_1}\right| > \frac{\varepsilon}{3}\hat{c_1}\right] + \Pr\left[\left|(1-\boldsymbol{F}_i^+)\cdot\Big(-\log(1-\boldsymbol{F}_0^+)\Big) - \hat{c_2}\right| > \frac{\varepsilon}{3}\hat{c_2}\right]\\ &\le 4\exp\left(-2\ell\frac{\varepsilon^2\alpha_{\min}^2}{81}\right) + 4\exp\left(-2m\ell\cdot t^2(\varepsilon/9)\right)\\ &\text{(using equations (A5) and (A6))}\end{aligned} \tag{A7}$$

Concentration of $\boldsymbol{D}_i$. Moving on to the denominator, $\boldsymbol{D}_i = \boldsymbol{F}_0^+ \cdot (-\log \boldsymbol{F}_0^+) + (1 - \boldsymbol{F}_0^+) \cdot (-\log(1 - \boldsymbol{F}_0^+))$. Focusing on the first term, $\boldsymbol{F}_0^+ \cdot (-\log \boldsymbol{F}_0^+)$, observe that $\mathbb{E}[\boldsymbol{F}_0^+] = \alpha_{\text{pool}}$. Again using Lemma A2 with $\boldsymbol{X} = \boldsymbol{F}_0^+$ and denoting $A_1 = \hat{b}_1 \overset{\text{def}}{=} \alpha_{\text{pool}} \cdot (-\log \alpha_{\text{pool}})$ we get,

$$\begin{aligned}
&\Pr\left[\left|\boldsymbol{F}_0^+ \cdot (-\log \boldsymbol{F}_0^+) - \hat{b_1}\right| > \varepsilon \hat{b_1}\right] \\
&\leq \Pr\left[\left|\boldsymbol{F}_0^+ - \alpha_{\text{pool}}\right| > \frac{\varepsilon}{3}\alpha_{\text{pool}}\right] + \Pr\left[\left|(-\log \boldsymbol{F}_0^+ - (-\log \alpha_{\text{pool}})\right| > \frac{\varepsilon}{3} \cdot (-\log \alpha_{\text{pool}})\right] \\
&\leq 2\exp\left(-2m\ell\frac{\varepsilon^2}{9}\alpha_{\text{pool}}^2\right) + 2\exp\left(-2m\ell \cdot t^2(\varepsilon/3)\right) \text{ (using Lemma A4 and (A3))} \\
&\leq 2\exp\left(-2m\ell\frac{\varepsilon^2}{9}\bar{\alpha}_{\text{pool}}^2\right) + 2\exp\left(-2m\ell \cdot t^2(\varepsilon/3)\right) \text{ (since } \alpha_{\text{pool}} \geq \bar{\alpha}_{\text{pool}})
\end{aligned}$$

Similarly, for the second term choosing $\boldsymbol{X} = (1 - \boldsymbol{F}_0^+)$ and denoting $A_2 = \hat{b}_2 \overset{\text{def}}{=} (1 - \alpha_{\text{pool}}) \cdot (-\log(1 - \alpha_{\text{pool}}))$ in Lemma A4 we get:

$$\begin{aligned}
&\Pr\left[\left|(1 - \boldsymbol{F}_0^+) \cdot (-\log(1 - \boldsymbol{F}_0^+)) - \hat{b_2}\right| > \varepsilon \hat{b_2}\right] \\
&\leq \Pr\left[\left|(1 - \boldsymbol{F}_0^+) - (1 - \alpha_{\text{pool}})\right| > \frac{\varepsilon}{3} \cdot (1 - \alpha_{\text{pool}})\right] + \Pr\left[\left|(-\log(1 - \boldsymbol{F}_0^+) - (-\log(1 - \alpha_{\text{pool}}))\right| > \frac{\varepsilon}{3} \cdot (-\log(1 - \alpha_{\text{pool}}))\right] \\
&= \Pr\left[\left|\boldsymbol{F}_0^+ - \alpha_{\text{pool}}\right| > \frac{\varepsilon}{3} \cdot (1 - \alpha_{\text{pool}})\right] + \Pr\left[\left|(-\log(1 - \boldsymbol{F}_0^+) - (-\log(1 - \alpha_{\text{pool}}))\right| > \frac{\varepsilon}{3} \cdot (-\log(1 - \alpha_{\text{pool}}))\right] \\
&\leq 2\exp\left(-2m\ell\frac{\varepsilon^2}{9}(1 - \alpha_{\text{pool}})^2\right) + 2\exp\left(-2m\ell \cdot t^2(\varepsilon/3)\right) \text{ (using Lemma A4 and equation (A4))} \\
&\leq 2\exp\left(-2m\ell\frac{\varepsilon^2}{9}\bar{\alpha}_{\text{pool}}^2\right) + 2\exp\left(-2m\ell \cdot t^2(\varepsilon/3)\right) \text{ (since } (1 - \alpha_{\text{pool}}) \geq \bar{\alpha}_{\text{pool}})
\end{aligned}$$

Combining the above two, choosing $\boldsymbol{X}_1 = \boldsymbol{F}_0^+ \cdot (-\log \boldsymbol{F}_0^+)$, $\boldsymbol{X}_2 = (1 - \boldsymbol{F}_0^+) \cdot (-\log(1 - \boldsymbol{F}_0^+))$, $a_1 = \hat{b}_1$ and $a_2 = \hat{b}_2$ in Lemma A1, we get:

$$\begin{aligned}
&\Pr\left[\left|\boldsymbol{D}_i - (\hat{b}_1 + \hat{b_2})\right| > \frac{\varepsilon}{3}(\hat{b}_1 + \hat{b_2})\right] \\
&\leq \Pr\left[\left|\boldsymbol{F}_0^+ \cdot (-\log \boldsymbol{F}_0^+) - \hat{b}_1\right| > \frac{\varepsilon}{3}\hat{b}_1\right] + \Pr\left[\left|(1 - \boldsymbol{F}_0^+) \cdot \left(-\log(1 - \boldsymbol{F}_0^+)\right) - \hat{b}_2\right| > \frac{\varepsilon}{3}\hat{b}_2\right] \qquad \text{(A8)} \\
&\leq 4\exp\left(-2m\ell\frac{\varepsilon^2}{81}\bar{\alpha}_{\text{pool}}^2\right) + 4\exp\left(-2m\ell \cdot t^2(\varepsilon/9)\right)
\end{aligned}$$

Concentration of $\boldsymbol{V}_i$. Now that we have expressions for the concentration of the numerator and denominator, we can discuss the concentration of the embedding score $\boldsymbol{V}_i$. Choosing $\boldsymbol{X}_i = \boldsymbol{N}_i$, $\boldsymbol{Y} = \boldsymbol{D}_i$, $a_1 = \hat{c_1} + \hat{c_2}$, $b = \hat{b_1} + \hat{b_2}$ in Lemma A1, we get the required concentration bound for the uni-dimensional embedding of customer i:

$$\begin{aligned}
&\Pr\left[\left|\boldsymbol{V}_i - \frac{\hat{c_1} + \hat{c_2}}{\hat{b}_1 + \hat{b}_2}\right| > \varepsilon\frac{\hat{c_1} + \hat{c_2}}{\hat{b}_1 + \hat{b}_2}\right] \\
&= \Pr\left[\left|\frac{\boldsymbol{N}_i}{\boldsymbol{D}_i} - \frac{\hat{c_1} + \hat{c_2}}{\hat{b}_1 + \hat{b}_2}\right| > \varepsilon\frac{\hat{c_1} + \hat{c_2}}{\hat{b}_1 + \hat{b}_2}\right]
\end{aligned}$$

$$
\begin{aligned}
&\leq \Pr\left[|\boldsymbol{N}_i - (\hat{c_1}+\hat{c_2})| > \frac{\varepsilon}{3}\cdot(\hat{c_1}+\hat{c_2})\right] + \Pr\left[\left|\boldsymbol{D}_i - (\hat{b_1}+\hat{b_2})\right| > \frac{\varepsilon}{3}\cdot(\hat{b_1}+\hat{b_2})\right]\\
&\leq 4\exp\left(-2\ell\frac{\varepsilon^2\alpha_{\min}^2}{81}\right) + 4\exp\left(-2m\ell\frac{\varepsilon^2}{81}\bar{\alpha}_{\text{pool}}^2\right) + 8\exp\left(-2m\ell\cdot t^2(\varepsilon/9)\right)\\
&\text{(from equations (A7) and (A8))}\\
&\leq 4\exp\left(-2\ell\frac{\varepsilon^2\alpha_{\min}^2}{81}\right) + 12\exp\left(-2m\ell\cdot t^2(\varepsilon/9)\right)\\
&\left(\text{since } \frac{\log^2(1-\bar{\alpha}_{\text{pool}})}{(1-\log(1-\bar{\alpha}_{\text{pool}}))^2} < 1\right)
\end{aligned}
$$

Finally, note that $\hat{c_1}+\hat{c_2} = H(\alpha_{z_i},\alpha_{\text{pool}})$, the cross-entropy between the distributions $\text{Ber}(\alpha_{z_i})$ and $\text{Ber}(\alpha_{\text{pool}})$ and; $\hat{b_1}+\hat{b_2} = H(\alpha_{\text{pool}})$, the binary entropy function at α_{pool}. In other words, the uni-dimensional embedding score of customer i in segment k concentrates around the ratio $\frac{H(\alpha_k,\alpha_{\text{pool}})}{H(\alpha_{\text{pool}})}$ with high probability, as $\ell\to\infty$. □

Proof of Theorem 1. The result follows directly from the concentration of the embedding score to the ratio $\frac{H(\alpha_k,\alpha_{\text{pool}})}{H(\alpha_{\text{pool}})}$, refer to the discussion after Lemma 1 in the main text.

A.2. Asymptotic recovery of true segments

Having established the concentration of the customer embedding scores, we next discuss the error-rate of classification into the underlying segments. We begin by proving some useful lemmas. All notations are as stated in the main text, unless otherwise introduced.

Lemma A5. *Let k_1,k_2 be two arbitrary segments. Then for customer i, we have*

$$
\frac{|\boldsymbol{V}_i - H_{k_1}|}{H_{k_1}} \leq \frac{|H_{k_1}-H_{k_2}|}{2\cdot\max(H_{k_1},H_{k_2})} \implies \frac{|\boldsymbol{V}_i - H_{k_1}|}{H_{k_1}} \leq \frac{|\boldsymbol{V}_i - H_{k_2}|}{H_{k_2}}
$$

Proof. Consider the following:

$$
\begin{aligned}
\frac{|H_{k_1}-H_{k_2}|}{\max(H_{k_1},H_{k_2})} &= \frac{|(H_{k_1}-\boldsymbol{V}_i)+(\boldsymbol{V}_i-H_{k_2})|}{\max(H_{k_1},H_{k_2})}\\
&\leq \frac{|\boldsymbol{V}_i-H_{k_1}|}{\max(H_{k_1},H_{k_2})} + \frac{|\boldsymbol{V}_i-H_{k_2}|}{\max(H_{k_1},H_{k_2})}\\
&\text{(using triangle inequality)}\\
&\leq \frac{|\boldsymbol{V}_i-H_{k_1}|}{H_{k_1}} + \frac{|\boldsymbol{V}_i-H_{k_2}|}{H_{k_2}}\\
&\leq \frac{|H_{k_1}-H_{k_2}|}{2\cdot\max(H_{k_1},H_{k_2})} + \frac{|\boldsymbol{V}_i-H_{k_2}|}{H_{k_2}}\\
&\text{(follows from the hypothesis of the Lemma)}
\end{aligned}
$$

Therefore we have that

$$
\frac{|\boldsymbol{V}_i - H_{k_2}|}{H_{k_2}} \geq \frac{|H_{k_1}-H_{k_2}|}{2\cdot\max(H_{k_1},H_{k_2})} \geq \frac{|\boldsymbol{V}_i - H_{k_1}|}{H_{k_1}}
$$

□

Lemma A6. *Consider the constant Λ defined in Theorem 2:*

$$\Lambda = \frac{\left|\log \frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}}\right| \min_{k=1,2,\cdots,K}(\alpha_{k+1}-\alpha_k)}{2\left|\log \alpha_{\min}\right|}$$

Then, it follows that $\Lambda \le \min_{k\neq k'} \frac{\left|H_k - H_{k'}\right|}{2\cdot\max(H_k, H_{k'})} < 1$.

Proof. Recall that $H_k = \frac{H(\alpha_k,\alpha_{\text{pool}})}{H(\alpha_{\text{pool}})}$. Therefore, for any two segments $k \neq k'$, we have:

$$\Lambda_{kk'} \stackrel{\text{def}}{=} \frac{|H_k - H_{k'}|}{2\cdot \max(H_k, H_{k'})} = \frac{|H(\alpha_k,\alpha_{\text{pool}}) - H(\alpha_{k'},\alpha_{\text{pool}})|}{2\cdot\max\left\{H(\alpha_k,\alpha_{\text{pool}}), H(\alpha_{k'},\alpha_{\text{pool}})\right\}}$$

Next, observe that $H(\alpha_k,\alpha_{\text{pool}}) = -\alpha_k \log\frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}} - \log(1-\alpha_{\text{pool}})$, so that

$$|H(\alpha_k,\alpha_{\text{pool}}) - H(\alpha_{k'},\alpha_{\text{pool}})| = \left|\log\frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}}\right| |\alpha_k - \alpha_{k'}|$$

Now suppose $\alpha_{\text{pool}} > \frac{1}{2}$, this means that $H(\alpha_k,\alpha_{\text{pool}})$ is decreasing with α_k so that we have

$$\max\left\{H(\alpha_k,\alpha_{\text{pool}}), H(\alpha_{k'},\alpha_{\text{pool}})\right\} \le H(\alpha_{\min},\alpha_{\text{pool}}) \le H(\alpha_{\min}, 1-\alpha_{\min}) \le -\log\alpha_{\min}$$

where the second inequality follows from the fact that $\alpha_{\text{pool}} \le 1-\alpha_{\min}$ and the last inequality from the fact that $-\log(1-\alpha_{\min}) \le -\log\alpha_{\min}$. Similarly, when $\alpha_{\text{pool}} < \frac{1}{2}$, we have

$$\max\left\{H(\alpha_k,\alpha_{\text{pool}}), H(\alpha_{k'},\alpha_{\text{pool}})\right\} \le H(1-\alpha_{\min},\alpha_{\text{pool}}) \le H(1-\alpha_{\min},\alpha_{\min}) = H(\alpha_{\min},1-\alpha_{\min}) \le -\log\alpha_{\min}$$

where the second inequality is true because $\alpha_{\text{pool}} \ge \alpha_{\min}$.

Combining the above observations and using the fact that $|\alpha_k - \alpha_{k'}| \ge \min_{k''=1,2,\ldots,K-1}(\alpha_{k''+1} - \alpha_{k''})$ for all $k \neq k'$, we get $\Lambda_{kk'} \ge \Lambda$ for all $k \neq k'$. Further, observe that $\Lambda_{kk'} < 1$ because $H_k > 0$ for all $1 \le k \le K$. Therefore, $\Lambda \le \min_{k\neq k'}\Lambda_{kk'} < 1$. □

Lemma A7. *Consider customer i and suppose we have the following:*

$$\frac{|\boldsymbol{V}_i - H_{z_i}|}{H_{z_i}} \le \frac{|H_{z_i} - H_{k'}|}{2\cdot\max(H_{z_i}, H_{k'})} \quad \forall\, k' \neq z_i$$

Then it follows that $\hat{\boldsymbol{I}}(i) = z_i$, i.e we correctly classify customer i. Conversely, we have

$$\Pr\left[\hat{\boldsymbol{I}}(i) \neq z_i\right] \le \Pr\left[|\boldsymbol{V}_i - H_{z_i}| > \Lambda\cdot H_{z_i}\right]$$

Proof. Using Lemma A5 we obtain that $\frac{\left|\boldsymbol{V}_i - H_{z_i}\right|}{H_{z_i}} \le \frac{\left|\boldsymbol{V}_i - H_{k'}\right|}{H_{k'}}$ for all $k' \neq z_i$. This means that $\arg\min_{k\in[K]} \frac{|\boldsymbol{V}_i - H_k|}{H_k} = z_i$. For the second part of the claim, observe that if $\hat{\boldsymbol{I}}(i) \neq z_i$ then there exists some $k \neq z_i$ such that $\frac{\left|\boldsymbol{V}_i - H_{z_i}\right|}{H_{z_i}} > \frac{\left|H_{z_i} - H_k\right|}{2\cdot\max(H_{z_i}, H_k)} \ge \Lambda$, which follows from Lemma A6 above. In other words,

$$\hat{\boldsymbol{I}}(i) \neq z_i \implies |\boldsymbol{V}_i - H_{z_i}| > \Lambda\cdot H_{z_i}$$

and the claim follows. □

Proof of Theorem 2. The probability that customer i is misclassified by the nearest-neighbor classifier $\hat{\boldsymbol{I}}(\cdot)$ is given by:

$$\begin{aligned}
\Pr\left[\hat{\boldsymbol{I}}(i) \neq z_i\right] &\leq \Pr\left[|\boldsymbol{V}_i - H_{z_i}| > \Lambda \cdot H_{z_i}\right] \text{ (using Lemma A7)} \\
&\leq 4\exp\left(-2\ell\frac{\Lambda^2\alpha_{\min}^2}{81}\right) + 12\exp\left(\frac{-2m\cdot\ell\cdot\Lambda^2\bar{\alpha}_{\text{pool}}^2\log^2(1-\bar{\alpha}_{\text{pool}})}{81\big(1-\log(1-\bar{\alpha}_{\text{pool}})\big)^2}\right) \\
&\text{(using result of Lemma 1)}
\end{aligned}$$

Now given some $0 < \delta < 1$, suppose that the number of observations from each customer satisfy:

$$\ell \geq \frac{648}{\lambda^2}\cdot\left(\frac{\log\alpha_{\min}}{\log(1-\alpha_{\min})\cdot\alpha_{\min}}\right)^2\cdot\frac{1}{\log^2\frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}}}\cdot\log(16/\delta)$$

Then, it follows from above that

$$\begin{aligned}
\Pr\left[\hat{\boldsymbol{I}}(i) \neq z_i\right] &\leq 4\exp\left(-2\ell\frac{\Lambda^2\alpha_{\min}^2}{81}\right) + 12\exp\left(\frac{-2m\cdot\ell\cdot\Lambda^2\bar{\alpha}_{\text{pool}}^2\log^2(1-\bar{\alpha}_{\text{pool}})}{81\big(1-\log(1-\bar{\alpha}_{\text{pool}})\big)^2}\right) \\
&\leq 4\exp\left(-2\ell\frac{\Lambda^2\alpha_{\min}^2}{81}\right) + 12\exp\left(\frac{-2m\cdot\ell\cdot\Lambda^2\alpha_{\min}^2\log^2(1-\alpha_{\min})}{81\big(1-\log(1-\alpha_{\min})\big)^2}\right) \\
&\text{(since } \bar{\alpha}_{\text{pool}} \geq \alpha_{\min}) \\
&\leq 4\exp\left(-2\ell\frac{\Lambda^2\alpha_{\min}^2\log^2(1-\alpha_{\min})}{81\left(1-\log(1-\alpha_{\min})\right)^2}\right) + 12\exp\left(\frac{-2m\cdot\ell\cdot\Lambda^2\alpha_{\min}^2\log^2(1-\alpha_{\min})}{81\big(1-\log(1-\alpha_{\min})\big)^2}\right) \\
&\left(\text{since } \log^2(1-\alpha_{\min}) < \left(1-\log(1-\alpha_{\min})\right)^2\right) \\
&\leq 16\exp\left(-2\ell\frac{\Lambda^2\alpha_{\min}^2\log^2(1-\alpha_{\min})}{81\left(1-\log(1-\alpha_{\min})\right)^2}\right) \\
&\text{(since } m \geq 1) \\
&= 16\exp\left(-\ell\frac{\lambda^2\alpha_{\min}^2\log^2\frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}}\log^2(1-\alpha_{\min})}{162\cdot\log^2\alpha_{\min}\cdot\left(1-\log(1-\alpha_{\min})\right)^2}\right) \\
&\text{(substituting the value of } \Lambda) \\
&\leq 16\exp\left(-\ell\frac{\lambda^2\alpha_{\min}^2\log^2\frac{\alpha_{\text{pool}}}{1-\alpha_{\text{pool}}}\log^2(1-\alpha_{\min})}{648\cdot\log^2\alpha_{\min}}\right) \\
&\left(\text{since } \alpha_{\min} < \frac{1}{2} \implies 1-\log(1-\alpha_{\min}) < 2\right) \\
&\leq \delta \\
&\left(\text{using the bound on } \ell\right)
\end{aligned}$$

Next, suppose that $m\cdot\frac{\log^2(1-\bar{\alpha}_{\text{pool}})}{\big(1-\log(1-\bar{\alpha}_{\text{pool}})\big)^2} \geq 1$, and observe that $\bar{\alpha}_{\text{pool}} \geq \alpha_{\min}$. Then we get,

$$\Pr\left[\hat{\boldsymbol{I}}(i) \neq z_i\right] \leq \Pr\left[|\boldsymbol{V}_i - H_{z_i}| > \Lambda \cdot H_{z_i}\right] \text{ (using Lemma A7)}$$

$$\leq 4\exp\left(-2\ell\frac{\Lambda^2\alpha^2_{\min}}{81}\right)+12\exp\left(\frac{-2m\cdot\ell\cdot\Lambda^2\bar{\alpha}^2_{\text{pool}}\log^2(1-\bar{\alpha}_{\text{pool}})}{81\big(1-\log(1-\bar{\alpha}_{\text{pool}})\big)^2}\right)$$

(using result of Lemma 1)

$$\leq 16\exp\left(-2\ell\frac{\Lambda^2\alpha^2_{\min}}{81}\right)$$

Substituting $\ell=\log n$ we get the desired result.

Appendix B: Latent Class Independent Category (LC-IND-CAT) model

Recall that segment k is characterized by B-dimensional vector $\boldsymbol{\alpha}_k$ such that α_{kb} represents the probability of liking any item $j\in\mathcal{I}_b$. Let $\boldsymbol{X}^+_{ib}$ denote the number of likes given by customer i for items in category b, i.e. $\boldsymbol{X}^+_{ib}=\sum_{j\in N_b(i)}\mathbf{1}[\boldsymbol{X}_{ij}=+1]$, where $N_b(i)\subset\mathcal{I}_b$ denotes the collection of items of category b rated by customer i. To calculate the embedding scores, we first need to compute the pooled estimate for each category. Since the underlying LC-IND-CAT model is parameterized by a vector of length B for each segment, the pooled estimate is given by $\bar{\boldsymbol{F}}^+_0=(\boldsymbol{F}^+_{01},\boldsymbol{F}^+_{02},\ldots,\boldsymbol{F}^+_{0B})$ where:

$$\boldsymbol{F}^+_{0b}\overset{\text{def}}{=}\frac{\sum_{i'=1}^m\boldsymbol{X}^+_{i'b}}{m\cdot\ell_b}\quad\forall\, b\in[B]$$

where recall that ℓ_b is the number of items in category b that each customer rates. Also, let us denote the fraction of likes given by customer i for category b items as $\boldsymbol{F}^+_{ib}\overset{\text{def}}{=}\frac{\boldsymbol{X}^+_{ib}}{\ell_b}$. We first establish a result that will be useful in the proof:

Lemma A8. *Given any $t>0$, for each category $b\in[B]$, the following facts are true:*

$$\text{(i)}\ \ \boldsymbol{X}^+_{ib}\sim\text{Bin}(\ell_b,\alpha_{z_ib})$$
$$\text{(ii)}\ \ \mathbb{E}[\boldsymbol{F}^+_{0b}]=\alpha_{b,\text{pool}}$$
$$\text{(iii)}\ \ \Pr\left[\left|\boldsymbol{F}^+_{0b}-\alpha_{b,\text{pool}}\right|\geq t\right]\leq 2\exp\left(-2m\cdot\ell_b\cdot t^2\right)$$

Proof. Lets begin with part (i). Observe that $\boldsymbol{X}^+_{ib}=\sum\limits_{j\in N_b(i)}\mathbf{1}[\boldsymbol{X}_{ij}=+1]$. Based on the generative model, we have that $\mathbf{1}[\boldsymbol{X}_{ij}=+1]$ are i.i.d. such that $\Pr[\boldsymbol{X}_{ij}=+1]=\alpha_{z_ib}$. The claim then follows.

For part (ii) observe that,

$$\mathbb{E}[\boldsymbol{F}^+_{0b}]=\frac{\sum_{i'=1}^m\mathbb{E}[\boldsymbol{X}^+_{i'b}]}{m\cdot\ell_b}=\frac{\sum_{i'=1}^m\ell_b\alpha_{z_{i'}b}}{m\cdot\ell_b}=\frac{\sum_{k'=1}^K(q_{k'}m)\cdot(\ell_b\alpha_{k'b})}{m\cdot\ell_b}=\sum_{k'=1}^K q_{k'}\alpha_{k'b}=\alpha_{b,\text{pool}}$$

using the fact that proportion $q_{k'}$ of the customer population belongs to segment k'. For part (iii), observe that $\boldsymbol{F}^+_{0b}$ can be written as:

$$\boldsymbol{F}^+_{0b}=\frac{\sum\limits_{i'=1}^m\sum_{j\in N_b(i)}\mathbf{1}[\boldsymbol{X}_{ij}=+1]}{m\cdot\ell_b}$$

In other words, $\boldsymbol{F}^+_{0b}$ is an average of $m \cdot \ell_b$ random variables, which are independent under the LC-IND-CAT model (since ratings for items within the same category are independent and the observations of different customers are generated independently). Then using Hoeffding's inequality we can show, for any $t > 0$:

$$\Pr\left[\left|\boldsymbol{F}^+_{0b} - \alpha_{b,\text{pool}}\right| \geq t\right] \leq 2\exp\big(-2m \cdot \ell_b \cdot t^2\big)$$

□

B.1. Concentration of customer embedding vectors

Proof of Lemma 2. For customer i that belongs to segment k, the embedding vector computed by our algorithm, $\overrightarrow{\boldsymbol{V}_i} = (\boldsymbol{V}_{i1}, \boldsymbol{V}_{i2}, \cdots, \boldsymbol{V}_{iB})$ where:

$$\begin{aligned}
\boldsymbol{V}_{ib} &= \frac{-\sum_{j \in N_b(i)} \big(\mathbf{1}[\boldsymbol{X}_{ij} = +1]\log \boldsymbol{F}^+_{0b} + \mathbf{1}[\boldsymbol{X}_{ij} = -1]\log(1 - \boldsymbol{F}^+_{0b})\big)}{-\sum_{j \in N_b(i)} \big(\boldsymbol{F}^+_{0b}\log \boldsymbol{F}^+_{0b} + (1 - \boldsymbol{F}^+_{0b})\log(1 - \boldsymbol{F}^+_{0b})\big)} \\
&= \frac{-\big(\sum_{j \in N_b(i)} \mathbf{1}[\boldsymbol{X}_{ij} = +1]\big)\log \boldsymbol{F}^+_{0b} - \big(\sum_{j \in N_b(i)} \mathbf{1}[\boldsymbol{X}_{ij} = -1]\big)\log(1 - \boldsymbol{F}^+_{0b})}{-\big(\boldsymbol{F}^+_{0b}\log \boldsymbol{F}^+_{0b} + (1 - \boldsymbol{F}^+_{0b})\log(1 - \boldsymbol{F}^+_{0b})\big) \cdot \ell_b} \\
&= \frac{-\big(\frac{\boldsymbol{X}^+_{ib}}{\ell_b}\big)\log \boldsymbol{F}^+_{0b} - \big(1 - \frac{\boldsymbol{X}^+_{ib}}{\ell_b}\big)\log(1 - \boldsymbol{F}^+_{0b})}{-\big(\boldsymbol{F}^+_{0b}\log \boldsymbol{F}^+_{0b} + (1 - \boldsymbol{F}^+_{0b})\log(1 - \boldsymbol{F}^+_{0b})\big)} \\
&= \frac{-\boldsymbol{F}^+_{ib}\log \boldsymbol{F}^+_{0b} - \big(1 - \boldsymbol{F}^+_{ib}\big)\log(1 - \boldsymbol{F}^+_{0b})}{-\big(\boldsymbol{F}^+_{0b}\log \boldsymbol{F}^+_{0b} + (1 - \boldsymbol{F}^+_{0b})\log(1 - \boldsymbol{F}^+_{0b})\big)}
\end{aligned}$$

Observe that the precise sequence of arguments given in the proof of Lemma 1 earlier can be repeated, for each item category b separately. More precisely, it follows that, given any $0 < \varepsilon < 1$, and for each $b \in [B]$:

$$\Pr\left[\left|\boldsymbol{V}_{ib} - \frac{H(\alpha_{z_i b}, \alpha_{b,\text{pool}})}{H(\alpha_{b,\text{pool}})}\right| > \varepsilon \frac{H(\alpha_{z_i b}, \alpha_{b,\text{pool}})}{H(\alpha_{b,\text{pool}})}\right] \leq 4\exp\left(-2\ell_b \frac{\varepsilon^2 \alpha^2_{\min}}{81}\right) + 12\exp\left(-2m \cdot \ell_b \cdot t^2_b(\varepsilon/9)\right)$$

where $t_b(\varepsilon) \overset{\text{def}}{=} \varepsilon \cdot \left(\frac{-\bar{\alpha}_{b,\text{pool}}\log(1 - \bar{\alpha}_{b,\text{pool}})}{1 - \log(1 - \bar{\alpha}_{b,\text{pool}})}\right)$ and $\bar{\alpha}_{b,\text{pool}} = \min\{\alpha_{b,\text{pool}}, 1 - \alpha_{b,\text{pool}}\}$.

Then, we consider the convergence of the vector $\overrightarrow{\boldsymbol{V}_i}$. Define the B-dimensional vector $\boldsymbol{H}_k = (H_{k1}, H_{k2}, \cdots, H_{kB})$ for each $k \in [K]$ such that $H_{kb} = \frac{H(\alpha_{kb}, \alpha_{b,\text{pool}})}{H(\alpha_{b,\text{pool}})}$, note that each $H_{kb} > 0$ (since all parameters are bounded), so that $\|\boldsymbol{H}_k\|_1 > 0$ for all $k \in [K]$. Then, using Lemma A1(iv) it follows that:

$$\begin{aligned}
\Pr\left[\left\|\overrightarrow{\boldsymbol{V}_i} - \boldsymbol{H}_{z_i}\right\|_1 > \varepsilon \left\|\boldsymbol{H}_{z_i}\right\|_1\right] &= \Pr\left[\sum_{b=1}^{B} |\boldsymbol{V}_{ib} - H_{z_i b}| > \varepsilon \cdot \left(\sum_{b=1}^{B} H_{z_i b}\right)\right] \\
&\leq \sum_{b=1}^{B} \Pr\left[|\boldsymbol{V}_{ib} - H_{z_i b}| > \varepsilon H_{z_i b}\right] \\
&\leq \sum_{b=1}^{B} 4\exp\left(-2\ell_b \frac{\varepsilon^2 \alpha^2_{\min}}{81}\right) + 12\exp\left(-2m \cdot \ell_b \cdot t^2_b(\varepsilon/9)\right) \\
&\leq 4 \cdot B \cdot \exp\left(-2\ell_{\min} \frac{\varepsilon^2 \alpha^2_{\min}}{81}\right) + 12 \cdot B \cdot \exp\left(-2m \cdot \ell_{\min} \cdot t^2_{\min}(\varepsilon/9)\right)
\end{aligned}$$

where $t_{\min}(\varepsilon) \overset{\text{def}}{=} \varepsilon \cdot \left(\frac{-\hat{\alpha}_{\text{pool}}\log(1-\hat{\alpha}_{\text{pool}})}{1-\log(1-\hat{\alpha}_{\text{pool}})}\right)$ and $\hat{\alpha}_{\text{pool}} = \min_{b\in[B]} \bar{\alpha}_{b,\text{pool}}$. The last inequality follows from the facts that $\ell_b \geq \ell_{\min}$ and $\bar{\alpha}_{b,\text{pool}} \geq \hat{\alpha}_{\text{pool}}$ for all $b \in [B]$. Substituting for $t_{\min}(\varepsilon)$ in the equation above establishes the result. $\square$

B.2. Asymptotic recovery of customer segments

We begin by stating analogous versions of Lemmas A5-A7.

Lemma A9. *Let k_1, k_2 be two arbitrary segments and $\|\cdot\|$ be an arbitrary norm on $\mathbb{R}^B$. Then for customer i, we have*

$$\frac{\left\|\overrightarrow{\boldsymbol{V}}_i - \boldsymbol{H}_{k_1}\right\|}{\|\boldsymbol{H}_{k_1}\|} \leq \frac{\|\boldsymbol{H}_{k_1} - \boldsymbol{H}_{k_2}\|}{2\cdot\max(\|\boldsymbol{H}_{k_1}\|, \|\boldsymbol{H}_{k_2}\|)} \implies \frac{\left\|\overrightarrow{\boldsymbol{V}}_i - \boldsymbol{H}_{k_1}\right\|}{\|\boldsymbol{H}_{k_1}\|} \leq \frac{\left\|\overrightarrow{\boldsymbol{V}}_i - \boldsymbol{H}_{k_2}\right\|}{\|\boldsymbol{H}_{k_2}\|}$$

Proof. The proof follows from a similar argument as in Lemma A5. $\square$

Lemma A10. *Consider the constant Γ defined in Theorem 4. Then it follows that $\Gamma \leq \min_{k'\neq k} \frac{\|\boldsymbol{H}_k - \boldsymbol{H}_{k'}\|}{2\cdot\max\left(\|\boldsymbol{H}_k\|, \|\boldsymbol{H}_{k'}\|\right)} < 1$.*

Proof. Recall that $\boldsymbol{H}_k \in \mathbb{R}^B$ such that $H_{kb} = \frac{H(\alpha_{kb},\alpha_{b,\text{pool}})}{H(\alpha_{b,\text{pool}})}$ for each $1 \leq b \leq B$. Therefore, we can write

$$\Gamma_{kk'} \overset{\text{def}}{=} \frac{\|\boldsymbol{H}_k - \boldsymbol{H}_{k'}\|}{2\cdot\max(\|\boldsymbol{H}_k\|, \|\boldsymbol{H}_{k'}\|)} = \frac{\sum_{b=1}^{B} \frac{\left|H(\alpha_{kb},\alpha_{b,\text{pool}}) - H(\alpha_{k'b},\alpha_{b,\text{pool}})\right|}{H(\alpha_{b,\text{pool}})}}{2\cdot\max(\|\boldsymbol{H}_k\|, \|\boldsymbol{H}_{k'}\|}\Bigg)$$

Next, observe that $H(\alpha_{kb}, \alpha_{b,\text{pool}}) = -\alpha_{kb}\log\frac{\alpha_{b,\text{pool}}}{1-\alpha_{b,\text{pool}}} - \log(1-\alpha_{b,\text{pool}})$, so that

$$|H(\alpha_{kb}, \alpha_{b,\text{pool}}) - H(\alpha_{k'b}, \alpha_{b,\text{pool}})| = \left|\log\frac{\alpha_{b,\text{pool}}}{1-\alpha_{b,\text{pool}}}\right| |\alpha_{kb} - \alpha_{k'b}|$$

Note that $H(\alpha_{b,\text{pool}}) \leq 1$ for any category b, using the definition of the binary entropy function. Therefore it follows,

$$\begin{aligned}
\sum_{b=1}^{B} \frac{|H(\alpha_{kb}, \alpha_{b,\text{pool}}) - H(\alpha_{k'b}, \alpha_{b,\text{pool}})|}{H(\alpha_{b,\text{pool}})} &= \sum_{b=1}^{B} \frac{\left|\log\frac{\alpha_{b,\text{pool}}}{1-\alpha_{b,\text{pool}}}\right| |\alpha_{kb} - \alpha_{k'b}|}{H(\alpha_{b,\text{pool}})} \\
&\geq \sum_{b=1}^{B} \left|\log\frac{\alpha_{b,\text{pool}}}{1-\alpha_{b,\text{pool}}}\right| |\alpha_{kb} - \alpha_{k'b}| \\
&\geq \gamma
\end{aligned}$$

where γ is as defined in the theorem. Next, consider $\|\boldsymbol{H}_k\|$ for some segment k:

$$\|\boldsymbol{H}_k\| = \sum_{b=1}^{B} \frac{H(\alpha_{kb}, \alpha_{b,\text{pool}})}{H(\alpha_{b,\text{pool}})} \leq \frac{1}{H_{\min}} \sum_{b=1}^{B} H(\alpha_{kb}, \alpha_{b,\text{pool}})$$

where $H_{\min} \overset{\text{def}}{=} H(\alpha_{\min})$. The above statement is true because $\alpha_{\min} \leq \alpha_{b,\text{pool}} \leq 1 - \alpha_{\min}$ and the binary entropy function is symmetric around $\frac{1}{2}$ so that $H(\alpha_{\min}) = H(1-\alpha_{\min})$, from which it follows

$H(\alpha_{b,\text{pool}}) \geq H_{\min}$ for any category b. Further since $\alpha_{\min} \leq \alpha_{kb} \leq 1-\alpha_{\min}$, we can use the argument from Lemma A6 to obtain $H(\alpha_{kb}, \alpha_{b,\text{pool}}) \leq |\log \alpha_{\min}|$ for all segments k and item categories b. This further implies that $\|\boldsymbol{H}_k\| \leq \frac{B\cdot|\log \alpha_{\min}|}{H_{\min}}$ for all segments $k \in [K]$. Finally observe that $H_{\min} = -\alpha_{\min}\log\alpha_{\min} - (1-\alpha_{\min})\log(1-\alpha_{\min}) \geq -\log(1-\alpha_{\min})$, so that $\|\boldsymbol{H}_k\| \leq \frac{B\cdot|\log \alpha_{\min}|}{H_{\min}} \leq \frac{B\cdot|\log \alpha_{\min}|}{|\log(1-\alpha_{\min})|}$.

Combining the above observations, we get that $\Gamma_{kk'} \geq \Gamma$ for all $k \neq k'$. Further, observe that $\Gamma_{kk'} < 1$ since the vectors $\boldsymbol{H}_k$ contain only non-negative entries. Therefore, $\Gamma \leq \min_{k\neq k'} \Gamma_{kk'} < 1$. □

Lemma A11. *Consider a customer i and suppose the following is true for an arbitrary choice of norm $\|\cdot\|$ on $\mathbb{R}^B$:*

$$\frac{\left\|\overrightarrow{\boldsymbol{V}}_i - \boldsymbol{H}_{z_i}\right\|}{\|\boldsymbol{H}_{z_i}\|} \leq \frac{\|\boldsymbol{H}_{z_i} - \boldsymbol{H}_{k'}\|}{2\cdot\max(\|\boldsymbol{H}_{z_i}\|, \|\boldsymbol{H}_{k'}\|)} \quad \forall\, k' \neq z_i$$

Then it follows that $\hat{\boldsymbol{I}}_2(i) = z_i$, i.e we correctly classify customer i. Conversely, we have

$$\Pr\left[\hat{\boldsymbol{I}}_2(i) \neq z_i\right] \leq \Pr\left[\left\|\overrightarrow{\boldsymbol{V}}_i - \boldsymbol{H}_{z_i}\right\| > \Gamma\cdot\|\boldsymbol{H}_{z_i}\|\right]$$

Proof. The proof follows from an identical argument as in Lemma A7, using the results of Lemmas A9 and A10 above. □

Proof of Theorem 4. The probability that a customer i is misclassified by the nearest-neighbor classifier $\hat{\boldsymbol{I}}_2(\cdot)$ is given by:

$$\begin{aligned}\Pr\left[\hat{\boldsymbol{I}}_2(i) \neq z_i\right] &\leq \Pr\left[\left\|\overrightarrow{\boldsymbol{V}}_i - H_{z_i}\right\|_1 > \Gamma\cdot\|\boldsymbol{H}_{z_i}\|_1\right] \quad \text{(using Lemma A11)}\\ &\leq 4\cdot B\cdot\exp\left(-2\ell_{\min}\frac{\Gamma^2\alpha_{\min}^2}{81}\right) + 12\cdot B\cdot\exp\left(\frac{-2m\cdot\ell_{\min}\cdot\Gamma^2\hat{\alpha}_{\text{pool}}^2\log^2(1-\hat{\alpha}_{\text{pool}})}{81\big(1-\log(1-\hat{\alpha}_{\text{pool}})\big)^2}\right)\\ &\quad\text{(follows from Lemma 2)}\end{aligned}$$

Now given some $0 < \delta < 1$, suppose that the number of observations from each customer satisfy:

$$\ell_{\min} \geq \frac{648B^2}{\gamma^2}\cdot\left(\frac{\log\alpha_{\min}}{\log^2(1-\alpha_{\min})\cdot\alpha_{\min}}\right)^2\log(16B/\delta)$$

Then, it follows from above that

$$\begin{aligned}\Pr\left[\hat{\boldsymbol{I}}_2(i) \neq z_i\right] &\leq 4\cdot B\cdot\exp\left(-2\ell_{\min}\frac{\Gamma^2\alpha_{\min}^2}{81}\right) + 12\cdot B\cdot\exp\left(\frac{-2m\cdot\ell_{\min}\cdot\Gamma^2\cdot\hat{\alpha}_{\text{pool}}^2\log^2(1-\hat{\alpha}_{\text{pool}})}{81\big(1-\log(1-\hat{\alpha}_{\text{pool}})\big)^2}\right)\\ &\leq 4\cdot B\cdot\exp\left(-2\ell_{\min}\frac{\Gamma^2\alpha_{\min}^2}{81}\right) + 12\cdot B\cdot\exp\left(\frac{-2m\cdot\ell_{\min}\cdot\Gamma^2\cdot\alpha_{\min}^2\log^2(1-\alpha_{\min})}{81\big(1-\log(1-\alpha_{\min})\big)^2}\right)\\ &\quad\text{(since } \hat{\alpha}_{\text{pool}} \geq \alpha_{\min}\text{)}\\ &\leq 4\cdot B\cdot\exp\left(-2\ell_{\min}\frac{\Gamma^2\alpha_{\min}^2\log^2(1-\alpha_{\min})}{81\,(1-\log(1-\alpha_{\min}))^2}\right) + 12\cdot B\cdot\exp\left(\frac{-2m\cdot\ell_{\min}\cdot\Gamma^2\alpha_{\min}^2\log^2(1-\alpha_{\min})}{81\big(1-\log(1-\alpha_{\min})\big)^2}\right)\end{aligned}$$

$$\left(\text{since } \log^2(1-\alpha_{\min}) < (1-\log(1-\alpha_{\min}))^2\right)$$
$$\leq 16 \cdot B \cdot \exp\left(-2\ell_{\min}\frac{\Gamma^2\alpha_{\min}^2\log^2(1-\alpha_{\min})}{81\,(1-\log(1-\alpha_{\min}))^2}\right)$$
$$(\text{since } m \geq 1)$$
$$= 16 \cdot B \cdot \exp\left(-2\ell_{\min}\frac{\gamma^2\cdot\log^2(1-\alpha_{\min})\cdot\alpha_{\min}^2\log^2(1-\alpha_{\min})}{324B^2\cdot\log^2\alpha_{\min}\cdot(1-\log(1-\alpha_{\min}))^2}\right)$$
$$(\text{substituting value of } \Gamma)$$
$$\leq 16 \cdot B \cdot \exp\left(-\ell_{\min}\frac{\gamma^2\cdot\log^2(1-\alpha_{\min})\cdot\alpha_{\min}^2\log^2(1-\alpha_{\min})}{648B^2\cdot\log^2\alpha_{\min}}\right)$$
$$\left(\text{since } \alpha_{\min} < \frac{1}{2} \implies 1-\log(1-\alpha_{\min}) < 2\right)$$
$$\leq \delta$$
$$\left(\text{using the bound on } \ell_{\min}\right)$$

Finally, suppose that $m \cdot \frac{\log^2(1-\hat{\alpha}_{\text{pool}})}{\left(1-\log(1-\hat{\alpha}_{\text{pool}})\right)^2} \geq 1$, and observe that $\hat{\alpha}_{\text{pool}} \geq \alpha_{\min}$. Then we get,

$$\Pr\left[\hat{\boldsymbol{I}}_2(i) \neq z_i\right] \leq \Pr\left[\left\|\overrightarrow{\boldsymbol{V}}_i - \boldsymbol{H}_{z_i}\right\|_1 > \Gamma\cdot\|\boldsymbol{H}_{z_i}\|_1\right] \quad (\text{using Lemma A11})$$
$$\leq 4\cdot B\cdot\exp\left(-2\ell_{\min}\frac{\Gamma^2\alpha_{\min}^2}{81}\right) + 12\cdot B\cdot\exp\left(\frac{-2m\cdot\ell_{\min}\cdot\Gamma^2\cdot\hat{\alpha}_{\text{pool}}^2\log^2(1-\hat{\alpha}_{\text{pool}})}{81\big(1-\log(1-\hat{\alpha}_{\text{pool}})\big)^2}\right)$$
$$(\text{follows from Lemma 2})$$
$$\leq 4\cdot B\cdot\exp\left(-2\ell_{\min}\frac{\Gamma^2\alpha_{\min}^2}{81}\right) + 12\cdot B\cdot\exp\left(-2\ell_{\min}\frac{\Gamma^2\alpha_{\min}^2}{81}\right)$$
$$= 16\cdot B\cdot\exp\left(-2\ell_{\min}\frac{\Gamma^2\alpha_{\min}^2}{81}\right)$$

Substituting $\ell_{\min} = \log n$ we get the desired result.

Appendix C: Computational study

C.1. EM Algorithm for LC method

Let $\boldsymbol{\Theta} = \left[q_1, q_2, \cdots, q_K, \alpha_1, \alpha_2, \cdots, \alpha_K\right]$ denote the set of all parameters (refer to the setup in section 4). The total number of parameters is therefore $K + K = 2\cdot K$. For ease of notation, we assume that the customer-item preference graph is complete but the EM algorithm can be immediately extended for the case of incomplete graphs. Let $\mathcal{D} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ be the observed rating vectors from the m customers. Then assuming that the vectors $\boldsymbol{x}_i$ are sampled i.i.d. from the population mixture distribution, the log-likelihood of the data can be written as:

$$\log\Pr[\mathcal{D}|\boldsymbol{\Theta}] = \sum_{i=1}^{m}\log\sum_{k=1}^{K}q_k\left(\prod_{j=1}^{n}\alpha_k^{\mathbf{1}[x_{ij}=+1]}(1-\alpha_k)^{\mathbf{1}[x_{ij}=-1]}\right) \tag{A9}$$

where $\mathbf{1}[\cdot]$ denotes the indicator function. The MLE for the parameters can be computed via the EM algorithm by introducing the latent variables corresponding to the true segment of each customer, which we denote by $\boldsymbol{z} = [z_1, z_2, \dots, z_m]$ where $z_i \in [K]$ denotes the true segment of customer i. The complete log-likelihood can then be written as

$$\log \Pr[\mathcal{D}, \boldsymbol{z} | \boldsymbol{\Theta}] = \sum_{i=1}^{m} \sum_{k=1}^{K} \mathbf{1}[z_i = k] \log \left(q_k \prod_{j=1}^{n} \alpha_k^{\mathbf{1}[x_{ij}=+1]} (1-\alpha_k)^{\mathbf{1}[x_{ij}=-1]} \right)$$

The EM algorithm executes the following two steps in each iteration:

- **E-step**: Given the data $\mathcal{D}$ and the current estimate of the parameters $\boldsymbol{\Theta}^{(t)}$, we compute the conditional expectation of the log-likelihood (w.r.t to the unknown customer segments $\boldsymbol{z}$) as

$$\mathbb{E}\left\{\log \Pr[\mathcal{D}, \boldsymbol{z} | \boldsymbol{\Theta}^{(t)}]\right\} = \sum_{i=1}^{m} \sum_{k=1}^{K} \gamma_{ik}^{(t)} \left\{ \log q_k^{(t)} + \sum_{j=1}^{n} \left(\mathbf{1}[x_{ij} = 1] \log \alpha_k^{(t)} + \mathbf{1}[x_{ij} = -1] \log(1 - \alpha_k^{(t)}) \right) \right\}$$

  Here $\gamma_{ik}^{(t)} = \Pr[z_i = k \mid \mathcal{D}, \boldsymbol{\Theta}^{(t)}]$ is the posterior probability of customer i's latent segment being equal to $k \in [K]$, conditioned on the observed ratings and the current model parameters. We can compute $\gamma_{ik}^{(t)}$ using Bayes theorem as follows

$$\gamma_{ik}^{(t)} \propto \Pr[\boldsymbol{x}_i \Big| z_i = k; \boldsymbol{\Theta}^{(t)}] \cdot \Pr[z_i = k \Big| \boldsymbol{\Theta}^{(t)}] = \frac{\prod_{j=1}^{n} (\alpha_k^{(t)})^{\mathbf{1}[x_{ij}=+1]} (1 - \alpha_k^{(t)})^{\mathbf{1}[x_{ij}=-1]} q_k^{(t)}}{\sum_{\ell=1}^{K} \prod_{j=1}^{n} (\alpha_\ell^{(t)})^{\mathbf{1}[x_{ij}=+1]} (1 - \alpha_\ell^{(t)})^{\mathbf{1}[x_{ij}=-1]} q_\ell^{(t)}}$$

- **M-step**: Based on the current posterior estimates of the customer segment memberships $\gamma_{ik}^{(t)}$ and the observed data $\mathcal{D}$, the model parameters are updated by maximizing $\mathbb{E}\Bigg[\log \Pr[\mathcal{D}, \boldsymbol{z} \mid \boldsymbol{\Theta}^{(t)}]\Bigg]$, which can be shown to be a lower bound on the data log-likelihood (eq A9). Equating the derivative of the expected conditional log-likelihood w.r.t q_k to zero (with the additional constraint that $\sum_{\ell=1}^{K} q_\ell = 1$), we get the parameter estimate for the next iteration

$$q_k^{(t+1)} = \frac{\sum_{i=1}^{m} \gamma_{ik}^{(t)}}{m} \quad \text{for } k \in [K]$$

  Similarly, for the parameters α_k we get the following expression

$$\alpha_k^{(t+1)} = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} \gamma_{ik}^{(t)} \mathbf{1}[x_{ij} = +1]}{\sum_{i=1}^{m} \sum_{j=1}^{n} \gamma_{ik}^{(t)}} \quad \text{for } k \in [K]$$

We repeat these two steps until convergence of the log-likelihood $\log \Pr[\mathcal{D} \mid \boldsymbol{\Theta}]$.

C.2. Implementation details and additional benchmarks

We imposed $Beta(2,2)$ prior on the parameters α_k and $Dir(1.5, 1.5, \dots, 1.5)$ prior on the parameters q_k in the LC method, to avoid numerical issues for sparse graphs. Further, since the log-likelihood objective is non-convex, the LC method is sensitive to the starting configuration and consequently, we run both the EM and SLSQP approaches 10 times with different random initializations and report the best outcome.

We also consider two additional distance-based clustering benchmarks that do not require any explicit generative model underlying the observed user ratings as input for clustering the customers: (1) k-medoids (de Hoon et al. 2004), a generalization of the popular k-means algorithm that can account for missing observations and (2) spectral clustering (Ng et al. 2002), which we refer to as SC, that forms a similarity graph between the data points and then partitions this graph, based on its spectral decomposition, into subgraphs to obtain the clusters. Both of these methods take as input an appropriate distance (or similarity) measure between any two observations; we used the standard *cosine distance* measure: for any two customers $i \neq i'$ with ratings $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$ (with $x_{ij} = 0$ if customer i did not rate item j), the cosine similarity is defined as:

$$\text{cossim}(i, i') \stackrel{\text{def}}{=} \frac{\sum_{j=1}^{n} x_{ij} \cdot x_{i'j}}{\sqrt{\sum_{j=1}^{n} x_{ij}^2} \cdot \sqrt{\sum_{j=1}^{n} x_{i'j}^2}}$$

The measure is termed cosine similarity because it can be viewed as the cosine of the angle between the vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$. If we observe $\sum_{j=1}^{n} x_{ij} \cdot x_{i'j}$, then note that this is the difference between the number of *agreements* (i.e. both rate +1 or −1) and *disagreements* (i.e. one rates +1 and other rates −1) between i and i' for commonly rated items. This is scaled by (square root of) the product of the number of items rated by each customer in the denominator. More the number of agreements, larger is the similarity between the customers. The cosine similarity lies between −1 and 1, a cosine similarity of 1 indicates perfect agreement and −1 indicates perfect disagreement. The cosine distance is given by $\text{cosd}(i, i') := 1 - \text{cossim}(i, i')$ which lies between 0 and 2.

Table 5 reports the performance of the two benchmarks. It is evident that they perform poorly, this is not surprising since they do not take into account any model structure for the observed ratings and are not able to capture similarity between customers. Further, both the benchmarks need to compute all pairwise distances between the customers which hampers their running times in practice.

Appendix D: MovieLens case study

D.1. Details of benchmarks

The LC method assumes that the population is comprised of K segments with proportion q_k and like probability α_k for segment $k \in [K]$. Then, it estimates the parameters by maximizing the log-likelihood of the observed ratings:

$$\max_{\substack{q_1, q_2, \ldots, q_K \\ \alpha_1, \ldots, \alpha_K}} \sum_{i=1}^{m} \log \left(\sum_{k=1}^{K} q_k \prod_{j \in N^{\text{train}}(i)} \alpha_k^{\mathbf{1}[r_{ij}=+1]} (1 - \alpha_k)^{\mathbf{1}[r_{ij}=-1]} \right) \text{ s.t. } \sum_k q_k = 1, q_k \geq 0, 0 \leq \alpha_k \leq 1 \;\forall\; k$$

Table 5 Percentage accuracy in recovering true segments for different parameter settings

K	$1-p$	k-medoids	SC	α-EMBED	% IMPROVEMENT		AVERAGE SPEEDUP (X)	
					over k-medoids	over SC	over k-medoids	over SC
5	0.0	40.1	45.4	98.7	146	117	35	43
	0.2	39.9	44.2	97.5	144	121		
	0.4	37.4	38.5	95.1	154	147		
	0.6	35.6	36.5	89.2	151	144		
	0.8	34.4	34.6	75.9	121	119		
7	0.0	30.7	33.4	92.1	200	176	27	36
	0.2	31.7	32.3	88.3	179	173		
	0.4	31.4	29.1	83.1	165	186		
	0.6	29.4	27.2	74.6	154	174		
	0.8	26.6	25.4	61.4	131	142		
9	0.0	23.3	26.8	80.8	247	201	22	35
	0.2	23.1	26.4	75.6	227	186		
	0.4	21.6	23.2	70.4	226	203		
	0.6	20.6	20.1	61.5	199	206		
	0.8	19.8	19.2	49.1	148	156		

The parameters are K—number of customer segments and $(1-p)$—sparsity of the preference graph. Each observation above is an average over 30 experimental runs. All improvements are statistically significant according to a paired samples t-test at 1% significance level

We use the EM algorithm described in Appendix C.1 to estimate the parameters. Let α_k^{LC}, $k=1,2,\ldots,K$, denote the segment parameters estimated by the LC method and let γ_{ik} denote the posterior probability of membership in segment k for user i. Then, the predicted rating for a new movie j_{new} is given by: $\hat{r}^{\text{LC}}_{ij_{\text{new}}}=+1$ if $\sum_{k=1}^{K}\gamma_{ik}\alpha_k^{\text{LC}}\geq 0.5$ else $\hat{r}^{\text{LC}}_{ij_{\text{new}}}=-1$.

The EB method assumes that the population is described by a prior distribution $G_{prior}(\cdot)$ over the parameter $0\leq\alpha\leq 1$ where α represents the probability of liking any item; and each individual i samples $\alpha_i\sim G_{prior}$ and uses α_i to generate ratings for the movies. Given the observed ratings, the parameters of $G_{prior}(\cdot)$ are estimated using a standard technique like maximum likelihood or method-of-moments. Then, for each individual i, we compute the posterior mean $\hat{\alpha}_i$ based on the estimated prior $\hat{G}_{prior}$, i.e. $\hat{\alpha}_i=\int_0^1\alpha\cdot\hat{G}_{post,i}(\alpha)\,d\alpha$ where $\hat{G}_{post,i}$ is the posterior distribution for α_i. Since $0\leq\alpha\leq 1$ and the ratings $r_{ij}\in\{+1,-1\}$ are binary, we assume the prior is a beta distribution $Beta(a,b)$ with $a,b>0$ and estimate the parameters a,b using the method-of-moments. Then, the posterior distribution for α_i is given by (since the beta distribution is a conjugate prior for the binomial distribution):

$$\hat{G}_{post,i}=Beta\left(a+\sum_{j\in N^{\text{train}}(i)}\mathbf{1}[r_{ij}=+1],\;b+\sum_{j\in N^{\text{train}}(i)}\mathbf{1}[r_{ij}=-1]\right)$$

where r_{ij} is the rating given by user i for movie j and $N^{\text{train}}(i)$ is the set of movies rated by user i. Consequently, we have that

$$\hat{\alpha}_i=\frac{a+\sum_{j\in N^{\text{train}}(i)}\mathbf{1}[r_{ij}=+1]}{a+b+|N^{\text{train}}(i)|}$$

and given a new movie j_{new}, we predict $\hat{r}^{\text{EB}}_{ij_{\text{new}}} = +1$ if $\hat{\alpha}_i \geq 0.5$ otherwise $\hat{r}^{\text{EB}}_{ij_{\text{new}}} = -1$.

For the distance-based clustering benchmarks, k-medoids and spectral clustering SC, after clustering the population, we estimate a separate parameter α_k for each segment k and predict ratings for new movies using the estimated parameters (as done for our approach α-EMBED).

Table 6 Additional benchmarks for new movie recommendation on MovieLens dataset

Genre	Accuracy			% Improvement	
	k-medoids	SC	α-EMBED	over k-medoids	over SC
Action (K=2)	30.7	44.9	56.4	83.7	25.6
Comedy (K=4)	37.7	45.0	58.4	54.9	29.8
Drama (K=4)	38.6	47.4	57.2	48.2	20.7

The accuracies for the distance-based clustering benchmarks are reported in Table 6. We see that they perform poorly, this is expected—these techniques rely on a well-defined similarity measure between data points which becomes difficult to determine when there are missing observations.

D.2. Robustness to number of segments K

Table 7 Aggregate accuracy as a function of K

Genre	K	SC	k-medoids	LC	α-EMBED
Action	2	44.9	30.7	51.2	**56.4**
	3	33.3	36.9	46.0	**46.3**
	4	38.5	41.1	**47.6**	41.8
	5	40.3	30.7	47.6	**54.4**
Comedy	2	37.7	37.7	54.8	**57.1**
	3	46.9	37.7	**51.3**	50.4
	4	45.0	37.7	51.8	**58.4**
	5	46.4	42.9	52.7	**56.0**
Drama	2	38.6	38.6	55.9	**57.0**
	3	48.3	38.6	52.5	**53.0**
	4	46.4	38.6	53.0	**57.2**
	5	47.4	38.6	53.4	**56.1**

The best performing algorithm for each value of K is highlighted in bold.

As mentioned in the main text, our approach can provide data-driven guidance for choosing the number of segments. However, the actual choice might vary based on the application and consequently, a standard way in the literature is to select K by using a validation set (which we do in our experiments). Table 7 reports the accuracy for the different methods[1] as a function of K. We observe that the performance of our algorithm is robust to the choice of K, and we outperform the benchmarks in all but two scenarios.

[1] Note that the EB method does not depend on the choice of K

D.3. Partially specified model for ratings

We discuss here the application of our segmentation approach when the model generating user ratings is only partially specified. We focus on movies in the three genres, action, comedy and drama. Since movies were tagged with more than one genre in the dataset, we only considered movies that had exactly one of these 3 genres; this left us with a total of 1861 movies and a population of 6040 users rating on these movies. We consider the generative model introduced in Defn. 2, here we have $B = 3$ categories and the vector $\boldsymbol{\alpha}_k = (\alpha_{k,\text{action}}, \alpha_{k,\text{comedy}}, \alpha_{k,\text{drama}})$. For each genre, we separately estimated the pooled parameters $\alpha_{b,\text{pool}}$ as described in the main text. Then, we apply Algorithm 4 to compute a 3-dimensional embedding vector for each user. The $\mathcal{V}$ matrix is incomplete, i.e., there are certain users that did not rate on movies in all 3 genres and therefore we compute a rank $r = 2$ factorization[2] of the $\mathcal{V}$ matrix. Then, we clustered the embedding vectors using the k-means algorithm into $K = 5$ segments, where the choice of K was tuned using a validation set similar to the case for individual genres discussed in the main text.

We compare our approach to a LC benchmark, and use an EM algorithm (similar to the one derived above in Appendix C.1) for estimating the model parameters. Table 8 reports the accuracies as well as statistics on the training and test datasets. As expected, segmenting the population provides significant improvements compared to the population model, which assumes that users have homogeneous preferences. Our approach still outperforms the LC benchmark, by upto 20% for the action genre and by 6.8% in aggregate. Further, even with the matrix factorization step, our approach is still $\sim 3\times$ faster as compared to EM.

Table 8 Training/test data statistics and rating prediction accuracies when choosing $K = 5$ segments

Genre	Train Data			Test Data		Accuracy			% Improvement	
	Users	Movies	Ratings	Movies	Ratings	POP	LC	α-EMBED	over POP	over LC
Action	5857	314	179K	29	352	32.6	47.9	57.5	76.4	20.0
Comedy	5972	723	256K	170	1931	38.2	53.3	57.1	49.5	7.1
Drama	5993	824	238K	367	4070	38.8	52.9	55.3	42.5	4.5
Aggregate	6040	1861	673K	566	6353	38.0	52.6	56.2	47.9	6.8

Appendix E: Details on benchmark segmentation approaches in eBay case study

eBay has access to various types of demographic information about its users. In addition, specific teams within eBay have developed techniques to segment the population based on their historical

[2] We used the ALS (Alternating Least Squares) class in `pyspark.mllib.recommendation` module, which is part of Apache Spark's python API, to compute the factorization: `https://spark.apache.org/docs/latest/api/python/pyspark.mllib.html#pyspark.mllib.recommendation.ALS`

(e.g., purchase) activity on the website. Based on our eBay collaborators' experience and familiarity with these data sources, we decided on using the following features to segment the user population:

- *Age Group*: Users were grouped into one of 4 age groups—18-24, 25-44, 45-64 and 65+.
- *Education*: Users were partitioned based on their qualification—*High School*, *College*, *Graduate School* and *Vocational/Tech*.
- *Income*: Users were grouped into one of 4 income ranges—< \$30k, \$30-\$50k, \$50-\$100k and \$100k+.
- *Gender*: Users were partitioned based on gender—male or female.
- *Recency, Frequency and Monetary value (RFM) segments*: Users were grouped into 5 segments derived using their purchase activity and money spent on the site.
- *Behavior-based segments*: Users were grouped into 6 segments based on the predominant type of activity (buying vs selling) on the site.

About 80-90% of the users on average in our dataset were associated with each of the above features. Given their dense coverage, we decided to segment the user population using the above features and then trained a separate logistic regression classifier for each segment. However, this approach did not result in big improvements in the AUC—the maximum percentage improvement over the population model for a single user segment across any of the above segmentations was 0.92% for the CSA category and 1.2% for HG. This suggested that such demographic features and coarse-grained behavior information were not able to capture heterogeneity in the user preferences and consequently, we decided to focus directly on the (fine-grained) click and purchase signals provided by the users for the purposes of segmentation. However, given the extreme sparsity of these signals (as discussed in the main text), most existing clustering techniques were infeasible. Our approach is designed for such settings and consequently, we used it to segment the user population.

Appendix F: Additional discussion on related machine learning literature

The relevant literature in machine learning on clustering is vast. Here, we supplement our discussion in Section 1.1 with additional details and references on some popular similarity/distance-based clustering techniques.

Similarity- or distance-based techniques fall into two categories: *partitional* and *hierarchical*. Perhaps the most popular partitional clustering algorithm is k-means that chooses a set of centroids and partitions the data points by assigning each point to its closest centroid. It has many variants and extensions including k-medians that uses the $\mathcal{L}_1$-norm instead of the standard Euclidean distance; k-medoids that can work with any similarity measure and therefore account for missing entries in the data points; kernel k-means (Dhillon et al. 2004) that projects data points to a higher dimensional space (to ensure better separation) before clustering, and so on. Spectral clustering aims to overcome some limitations of k-means (such as finding convex and isotropic clusters) by projecting data points onto a low-dimensional space—achieved via spectral decomposition of an affinity (or similarity) matrix between data points—and then clustering the resulting projections using k-means. The algorithm is widely used in practice; see Von Luxburg (2007) for a nice overview and Filippone et al. (2008) for a survey. Recently, there has been work on exploiting sparse representations for spectral clustering of data points in a high-dimensional space (Wright et al. 2010, Wu et al. 2014) to overcome the "curse of dimensionality" issue for the standard Euclidean distance.

Hierarchical clustering algorithms, on the other hand, build a hierarchy of nested partitions, with the partition at the top level containing a single cluster of all data points and the partition at the bottom level containing a single cluster for each data point, where the partition at each level is formed by either merging the nearest clusters (bottom-up or *agglomerative*) or dividing a cluster (top-down or *divisive*), at the previous level. It is a widely used data analysis and visualization tool and enables one to choose the most "natural" clustering from the hierarchy based on the application at hand. However, hierarchical clustering is very sensitive to the group similarity (or distance) function employed for merging or dividing clusters.

As mentioned in the main text, the above approaches assume that either the data points are vectors in the Euclidean space or there is a well-defined similarity measure between the data points. Both of these assumptions do not hold in our setting because of the many missing entries, diversity in the categorical observations, and the unstructured item space, lacking consistent feature representations. The *embed* step in our algorithm deals with these challenges by computing a low-dimensional representation for each customer. These representations can then be used as inputs to the above techniques for the purpose of obtaining the segments.