

Capturing Heterogeneity Among Consumers with Multi-taste Preferences

Liu Liu

NYU Stern School of Business, lliu@stern.nyu.edu,

Daria Dzyabura

NYU Stern School of Business, ddzyabur@stern.nyu.edu,

June 28, 2017

Please click [here](#) for the latest version

Previous research has suggested that an individual consumer may have multiple tastes within a given product category. Multi-taste preferences are likely present in categories characterized by large product attribute spaces and many diverse products, such as music, videos, restaurants, or books. Capturing heterogeneity among such multi-taste consumers requires new methods, as two consumers may share some tastes but not others. This is a different type of heterogeneity than that captured by existing models, such as mixed logit or latent class models, which estimate only one taste for each individual.

In this paper, we propose a model that allows for heterogeneity among consumers with multiple tastes, and an estimation procedure that scales to potentially very high dimensional attribute spaces. In a numerical study, we simulate consumers with multiple preferences and demonstrate the proposed algorithm accurately recovers parameters, whereas single-taste benchmark models underfit and lead to a misleading picture of individual level preferences and the population preference distribution. We then test the algorithm empirically on a data set of recipe choices. We show that the algorithm scales to a large parameter space, and that the model fits the data better than single-taste benchmarks. We also demonstrate that the model uncovers rich patterns of underlying heterogeneity, such as what the different tastes are, how many consumers have each taste, and which tastes tend to be more and less likely to occur in the same individual.

Key words: choice modeling, heterogeneity, machine learning, optimization, recommender systems

1. Introduction

In many product categories, an individual consumer may have multiple different sets of preferences, or tastes. Previous research has documented a “multiple ideal point” phenomenon in the context of brand preferences (Lee et al. (2002), DeSarbo et al. (2008)). This may occur due to context dependence, variety seeking, or different products meeting different goals. We extend this notion to preferences for product attributes. Consider, for example, consumer preferences for recipes (cooking), which is our empirical application in this paper. A consumer might enjoy cooking healthy meals using fish and chicken, many vegetables, little butter and oil; and using grilling or steaming as the cooking method. The same consumer may also like throwing dinner parties, for which she prefers recipes that scale well so they can be cooked in large portions and that most people tend to like. The consumer has two distinct (though not necessarily mutually exclusive) tastes, with different specific criteria for each. Other examples include music, videos, restaurants, and books, all of which are characterized by large attribute spaces and many diverse products.

To accurately capture such a consumer’s preferences, the model must have flexibility to allow for a single consumer to have a number of tastes, each with its own set of parameters. The presence of multiple tastes on the individual level also requires a different treatment of consumer heterogeneity. Accurately capturing how preferences are distributed in a population is a central problem in marketing (Allenby and Rossi (1998)). It helps firms design product lines, target individuals, and plan in-store assortments. To market their products effectively, firms need to be able to predict the optimal product configuration and understand the distribution of preferences: Are consumers all similar in terms of their preferred attributes, or are they more differentiated? Is the population segmented into groups of consumers that all have similar preferences within said segment? When each consumer has

multiple different tastes, two consumers can now be similar on one taste but different on others. Thus, if we want to accurately capture preferences for a population of consumers with multiple tastes, we need a different treatment of heterogeneity. Specifically, we have to understand (1) what different tastes exist in the population, (2) which consumers have which tastes, and (3) how heterogeneous individual preferences are within each taste.

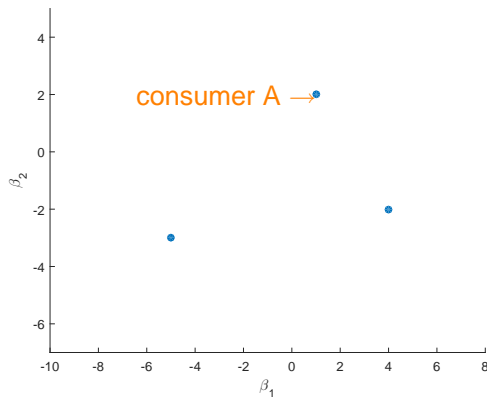
Capturing heterogeneity among multi-taste consumers is our goal in this paper. We formalize a population-level model of multi-taste consumers and propose an estimation procedure to estimate parameters. To allow for consumer heterogeneity, we use a hierarchical model in which each individual’s preferences contain a subset of population tastes and are heterogeneous within a single taste. This combinatorial property of the individual-taste subsets makes the objective function computationally expensive to estimate via the EM algorithm (see Section 4 and Appendix B for the derivation). We propose a computationally efficient iterative algorithm, which we call the Iterative Multi-taste Mixed Logit (IMML) for the hierarchical model, based on the CCCP algorithm (Section 4). We demonstrate in a set of simulation studies that the method accurately recovers parameter values. Importantly, our method scales to very large attribute spaces (84 attributes in our empirical application), which is essential because multiple tastes are most relevant in product categories defined over many attributes. For the empirical illustration we use recipe consumption data from a website on which users can browse recipes and select them by adding them to their “recipe box.” We demonstrate that the IMML method significantly outperforms single-taste benchmarks. We also uncover important patterns about the preference distribution: We identify four tastes, with most consumers having either two or three tastes. We also find that some tastes are much more likely to occur together than others. Comparing our parameter estimates with those of the single taste models,

we see huge differences among attributes (e.g., ingredients) that have positive weights in some tastes but negative weights in others. Butter is an example of such an attribute. Such attributes, unsurprisingly, are estimated to have a weight close to 0 by single taste models. These results highlight the importance of accounting for multi-taste preferences when characterizing the heterogeneity of preferences in a population.

The rest of this paper is organized as follows. Section 2 reviews related literature in marketing and computer science. In section 3, we formally specify the model. Section 4 describes our estimation approach. Section 5 describes results of a numerical study testing the algorithm, and the empirical application is in section 6. Section 7 concludes the paper.

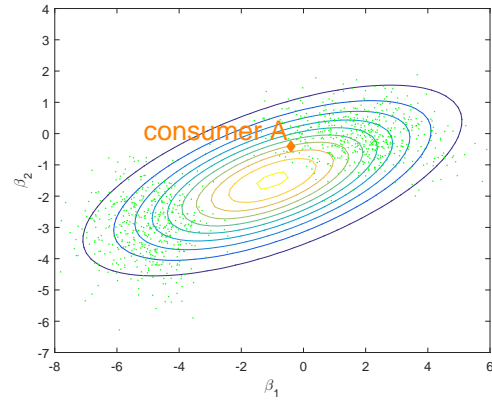
2. Related Work

In marketing, this paper is related to the literature on inferring consumer preferences from past purchase data, dating back to Guadagni and Little (1983). With the growing amount of consumer data available through digital channels, researchers have developed methods that are able to make accurate individual-level predictions for personalization on the internet (Yoganarasimhan (2015), Kim et al. (2007), Ying et al. (2006), Ansari et al. (2000), Ansari and Mela (2003), Bodapati (2008)). Researchers have used different approaches to capture heterogeneous preferences *across* consumers. For example, they have modeled preferences drawn from a finite mixture of latent classes (Kamakura and Russell (1989), Danaher and Mawhinney (2001), Bucklin and Gupta (1992), Chintagunta et al. (1991), Boxall and Adamowicz (2002)), as in Figure 1a, drawn from a continuous unimodal distribution (McFadden et al. (2000), Allenby and Rossi (1998), Rossi et al. (1996)), as in Figure 1b, and a mixture of continuous distributions (Greene and Hensher (2013)), as in Figure 1c. The current work differs from past research in that a single consumer can have more than one taste, as in Figure 1d, and two consumers can be similar on one taste but different on others.



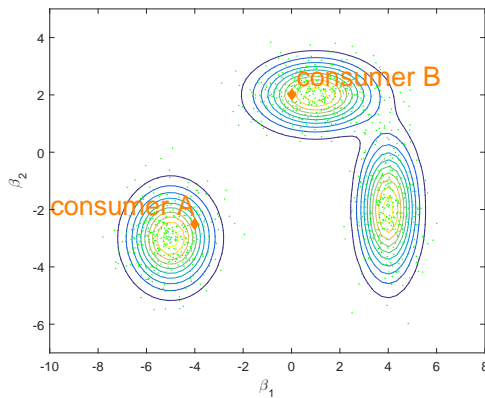
(a) Latent Class Model (Boxall and Adamowicz 2002).

Each consumer belongs to exactly one taste.

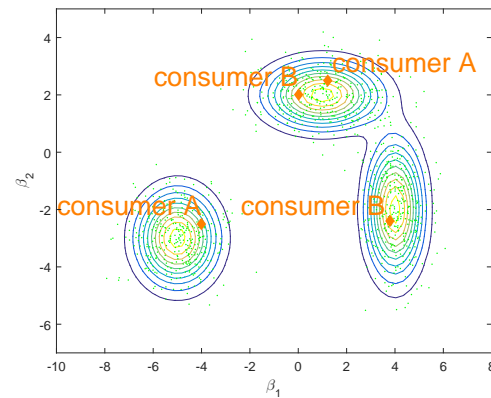


(b) Mixed Logit Model (McFadden and Train 2000).

All consumers' taste parameters are drawn from a uni-modal distribution.



(c) Latent Class Mixed Logit Model (Greene and Hensher 2013). Each consumer can and only can belong to one segment, though individual parameters can vary.



(d) Multi-taste consumer Model. Consumers can have more than one taste and belong to multiple segments.

Figure 1 Illustration of Consumer Heterogeneity Models

Studies have also captured preference *across* choice scenarios (i.e., along time) using first-order Markov processes or by modeling variety-seeking as a personal-trait parameter separately from taste parameters (Givon (1984), Chintagunta (1998), Varki and Chintagunta (2004)). However, all previous approaches assume that, in a specific choice scenario, each consumer has one taste and applies that taste to all products in her choice set. The

current work differs from this literature in that a single consumer may apply different choice criteria to different products even when they occur in the same choice set (simultaneously). Research on multi-category choice behavior captures the phenomenon that consumers have different tastes for different product categories and choose among products across multiple product categories in each shopping trip (Seetharaman et al. (2005)). Previous work models consumers' preference for different product categories jointly (Manchanda et al. (1999), Singh et al. (2005)), or considers consumers' choice of products from multiple categories a bundle choice (Chung and Rao (2003)). In our approach, consumers may have multiple tastes even within a single category, when all products share the same set of attributes. The complexity of the problem increases because the underlying subcategories are not known to the researcher a priori, and need to be learned simultaneously with the tastes.

Methodologically, this work is related to the growing literature of developing machine learning methods to study consumer preferences, both in preference elicitation (Huang and Luo (2012), Hauser et al. (2010), Dzyabura and Hauser (2011), Evgeniou et al. (2007), Evgeniou et al. (2005)) and in secondary data such as clicks (Yoganarasimhan (2015)) and text from user-generated content, such as reviews and forums (Tirunillai and Tellis (2014), Netzer et al. (2012)).

This paper also draws on literature in computer science, specifically, recommendation systems and machine learning. The problem of predicting a user's preferences for unrated items based on past behavior is well studied in recommendation systems. Most recommendation systems use collaborative filtering (CF) methods to produce user-specific recommendations based on patterns of ratings or purchases across all consumers (collaborating) (see Koren and Bell (2011) for a review). For example, matrix factorization methods SVD and SVD++ map both users and items to a joint latent factor space. In this latent space,

each item and each user is represented as an m -dimensional vector. This paper differs from this literature in that we explicitly model consumer choice behavior and obtain parameter estimates that correspond to product attributes. We draw on the machine-learning literature to design an efficient algorithm to estimate our model (Yuille and Rangarajan (2003), Felzenszwalb et al. (2008), Yu and Joachims (2009)).

3. Model

We now formally set up the model for multi-taste consumers. We begin by defining utility for a multi-taste consumer, then present the population-level model, and finally discuss two related models from the literature.

Utility. For a single-taste consumer, the utility of choosing product j is given by

$$U_j = \beta x_j + \varepsilon_j, j = 1, \dots, J, \quad (1)$$

where β is the consumer's taste, that is, a vector of weights on product attributes, x_j is a vector of product attributes, and $\varepsilon_j \sim$ i.i.d extreme value is the idiosyncratic error.

A multi-taste consumer has a set of K tastes $\{\beta_1, \dots, \beta_K\}$. Within each taste, the utility of a product j is linear in attributes, as in the single-taste consumer case. But across all tastes, we define utility as the maximum of utilities from multiple tastes:

$$U_j = \max_{k=1, \dots, K} \beta_k x_j + \varepsilon_j, j = 1, \dots, J. \quad (2)$$

If we observe that a consumer purchased a product, we can infer this product must have high utility on at least one of his/her tastes. We use max to capture this inference. A similar approach of using max-based nonlinearities of latent factors can be found in Weston et al. (2013) and Felzenszwalb et al. (2008) in machine learning. For such models, algorithms exist with proven guarantees for convergence, which we describe in more detail in the following estimation section.

The choice probability of choosing product j in a binary choice scenario is:

$$\Pr(j) = \frac{\exp\left(\max_{k=1,\dots,K} \beta_k x_j\right)}{1 + \exp\left(\max_{k=1,\dots,K} \beta_k x_j\right)}. \quad (3)$$

Let $Y = \{y_1, \dots, y_J\}$ be the choice decision of this consumer regarding all the products, and $y_j = 1$ if product j is chosen. The log-likelihood is

$$\ell(\{\beta_1, \beta_2, \dots, \beta_K\}) = \sum_{j=1}^J \left[y_j \max_{k=1,\dots,K} \beta_k x_j - \max_{k=1,\dots,K} \log(1 + \exp(\beta_k x_j)) \right]. \quad (4)$$

Heterogeneity. For multi-taste consumers, two sources of heterogeneity exist: first, as a random-effects model, within each taste, individual parameters are drawn from a population distribution; second, individuals also possess different sets of tastes.

Suppose a total of K tastes exist in population, and each taste follows a normal distribution $N(b_k, \Sigma_k)$, $k = 1, \dots, K$. Assume each taste membership follows a Bernoulli distribution with parameter α_k , $k = 1, \dots, K$. Each consumer i , $i = 1, \dots, I$ draws his tastes and makes choices as follows:

1. For each taste k , consumer i decides whether to possess it with probability α_k . Consumer i ends with a taste subset \mathbb{K}_i .
2. For each taste in \mathbb{K}_i , consumer i draws his/her individual parameter from the population taste distribution, namely,

$$\beta_{ik} \sim N(b_k, \Sigma_k), \forall k \in \mathbb{K}_i.$$

3. Conditional on the sampled K_i individual tastes $\{\beta_{ik}, \forall k \in \mathbb{K}_i\}$, consumer i makes choices by maximizing utilities using the multi-taste utility function defined in the section above. We consider a binary choice scenario. For all J products, consumer i makes a

decision to buy or not to buy. Let $Y_i = \{y_{i1}, \dots, y_{iJ}\}$ be the choice decision. Given consumer i 's set of tastes $\{\beta_{ik}, \forall k \in \mathbb{K}_i\}$, the probability of Y_i is

$$Q_i(\{\beta_{ik}, \forall k \in \mathbb{K}_i\}) = \prod_j \mathcal{L}_{ij}(\{\beta_{ik}, \forall k \in \mathbb{K}_i\}), \quad (5)$$

where \mathcal{L}_{ij} is the likelihood of choice of product j :

$$\mathcal{L}_{ij}(\{\beta_{ik}, k \in \mathbb{K}_i\}) = \left(\frac{\exp\left(\max_{k \in \mathbb{K}_i} \beta_{ik} x_j\right)}{1 + \exp\left(\max_{k \in \mathbb{K}_i} \beta_{ik} x_j\right)} \right)^{y_{ij}} \left(\frac{1}{1 + \exp\left(\max_{k \in \mathbb{K}_i} \beta_{ik} x_j\right)} \right)^{1-y_{ij}}. \quad (6)$$

Conditional on the model parameters $\theta = \{b_1, \Sigma_1, \dots, b_K, \Sigma_K, \alpha_1, \dots, \alpha_K\}$, the choice probability of consumer i is

$$P_i(\theta) = \sum_{\mathbb{K}_i \subseteq \mathbb{K}} s_{\mathbb{K}_i} \int \cdots \int Q_i(\{\beta_{ik}, \forall k \in \mathbb{K}_i\}) \left(\prod_{k \in \mathbb{K}_i} \phi(\beta_{ik} | b_k, \Sigma_k) \right) d\beta_{ik_1} \dots d\beta_{ik_{|\mathbb{K}_i|}}, \quad (7)$$

where $s_{\mathbb{K}_i}$ is the share of the population that has tastes subset \mathbb{K}_i ,

$$s_{\mathbb{K}_i} = \prod_{k \in \mathbb{K}_i} \alpha_k \prod_{k' \notin \mathbb{K}_i} (1 - \alpha_{k'}), \quad (8)$$

and $\phi(\beta_{ik} | b_k, \Sigma_k)$ is normal density with mean b_k and variance Σ_k . $\{k_1, \dots, k_{|\mathbb{K}_i|}\}$ is the taste subset \mathbb{K}_i .

The joint loglikelihood of the data of all consumers is:

$$\ell(\theta) = \sum_i \log P_i(\theta). \quad (9)$$

Related models. Two alternative ways to capture multiple tastes are worth mentioning. The first way is with a nested logit model (e.g., Kannan and Wright (1991), Guadagni and Little (1998), Chintagunta (1992)). The challenge, however, is that we do not know the “nests” corresponding to each product. The second way is to increase the dimensionality of product representation by allowing any interaction between product attributes, and to learn

consumers’ utility weights for all such interactions. This approach captures the nonlinearity of consumer preference on product attributes as in our model (by using “max”). However, it needs to figure out which set of interaction terms to add to the model; allowing all possible interactions leads the parameter space to grow exponentially. The high-dimensional model also takes longer to train and is likely to overfit. Regularization can be used to produce a sparse model, by performing feature selection while learning and penalizing extreme parameter values (e.g., Zou and Hastie (2005)). But regularization itself is time consuming, especially when dimensionality is high and when for allowing heterogeneity. Our model is both parsimonious and interpretable.

Next, we discuss the details of estimating the multi-taste consumer model.

4. Estimation

We first describe an iterative algorithm based on CCCP (Yuille and Rangarajan (2003)) for estimating the homogeneous multi-taste consumer model. It is simpler to estimate and also provides intuition of the algorithm for the heterogeneous multi-taste consumers case. We then describe the iterative algorithm we use to estimate the heterogeneous consumers case, called Iterative Multi-taste Mixed Logit (IMML).

4.1. CCCP for Homogeneous Multi-taste Consumer Model

In the homogeneous multi-taste case, all the consumers have the same set of multiple tastes $\{\beta_1, \beta_2, \dots, \beta_K\}$. Our goal is to find a set of parameters that maximize the log-likelihood function (or minimize the negative log-likelihood) given in equation 4, namely,

$$\{\beta_1^*, \beta_2^*, \dots, \beta_K^*\} = \arg \min_{\{\beta_1, \beta_2, \dots, \beta_K\}} \left[\underbrace{\sum_{j=1}^n \max_k \log(1 + e^{\beta_k x_j})}_{\text{convex}} - \underbrace{\sum_{j=1}^n y_j \max_k \beta_k x_j}_{\text{concave}} \right]. \quad (10)$$

A key property of the optimization function is that it’s a sum of a convex and a concave function; hence, we can solve the optimization problem using the CCCP (Yuille and Rangarajan (2003)). The general template for a CCCP algorithm for minimizing a function

$f(\beta) - g(\beta)$, where f and g are convex, iterates between two steps: (1) upper bound the concave part ($-g$) with a hyperplane, so that the resulting function is convex; and (2) minimize the resulting convex sum. A CCCP-based algorithm is guaranteed to decrease the objective function at every iteration and to converge to a local minimum or saddle point (Yuille and Rangarajan (2003)).

The CCCP algorithm for homogeneous multi-taste-consumers is listed in Algorithm 2 (see Appendix A). Line 3 in the algorithm is the step for computing the upper bound for the concave part $-\sum_{j=1}^n y_j \max_k \beta_k x_j$. The hyperplane constructed is $-\sum_{j=1}^n y_j \beta_{k_j^*}^T x_j$, where k_j^* is the index of the taste that gives the highest utility for product j . In fact, for the concave part, we only need to impute the taste index k_j^* for positive examples (i.e., $\forall j, y_j > 0$). The algorithm iterates between imputing the index of the taste, which gives the highest utility for each product that is purchased (i.e., positive instances), and solving the resulting convex sum while treating those taste indices as given. It is similar to algorithms used for learning SVM with latent variables (Felzenszwalb et al. (2008); Yu and Joachims (2009)).

4.2. IMML for Heterogeneous Multi-taste Consumer Model

To estimate taste parameters of heterogeneous multi-taste consumers, we propose the IMML algorithm. Appendix 1 derives an expectation-maximization (EM) algorithm to estimate $\theta = \{b_1, \Sigma_1, \dots, b_K, \Sigma_K, \alpha_{i1}, \dots, \alpha_{iK}\}$ by solving the maximum likelihood function:

$$\arg \max_{\theta} \sum_i \log P_i(\theta).$$

Though the maximization function is much simpler than the original log-likelihood function and we derive a closed form of the update function for the parameters, it's computationally expensive due to the combinatorial property of the individual-taste subsets and simulation draws. For each EM iteration, the time complexity is $O(2^K IRJK)$, in which I is the

number of consumers, J is the number of products, R is the number of draws per iteration, and K is the number of tastes in the population. Hence, instead, we propose the IMML to solve the problem efficiently.

The general approach of this algorithm is similar to the CCCP algorithm for the homogeneous multi-taste consumer case. It iterates between two steps: (1) given the individual taste parameters, for each person, impute the index of taste which gives the highest utility for each product that this consumer chose (i.e., positive instances); and (2) treating the taste index as given, learn model parameters and each individual's taste parameters. Specifically, the first step clusters products for each individual. For consumer i , suppose she has a taste subset \mathbb{K}_i . We cluster the products she purchased to the taste that gives the highest utility. We put copies of the products i didn't purchase into each of his taste clusters. The intuition is that if a product is not purchased, then it must have low utility on all of this consumer's tastes; hence we consider it a negative instance for any taste. After the first step, for each consumer and each product (i.e., each observation), we assign the product to a taste cluster. In the second step, for each taste cluster, based on the data assigned to it, we estimate a mixed logit model, assuming the individual parameters of all consumers who have the taste are drawn from a normal distribution. The algorithm continues until the products assignment for each consumer is the same as the previous iteration or until the objective function cannot be decreased below tolerance. If we use a similar EM-based approach to estimate the mixed logit model, the time complexity of each iteration is $O(IJKR)$. The pseudocode is *procedure* ITERATIVESTEP in Algorithm 1.

The initialization of the individual taste parameters are important. The individual taste parameters include the tastes subsets and individual utility weights associated with each taste in his taste subset. We utilize the heterogeneity of purchased products and clustering

Algorithm 1 The IMML algorithm for heterogeneous multi-taste consumer

Input: $\mathcal{D} = \{(x_1, y_1), \dots, (x_J, y_J)\}$ // Consumer's choice on all J products

 K // Number of tastes

 ϵ // Error threshold

Output: $\theta = \{b_1, \Sigma_1, \dots, b_K, \Sigma_K\}$

```

1: procedure INITIALIZATIONSTEP
2:   for each consumer  $i = 1, \dots, I$  do
3:     Cluster  $i$ 's purchased products to  $K_i$  clusters by product attributes
4:     Assign products that are not purchased to all  $K_i$  clusters
5:     for each individual product cluster  $k = 1, \dots, K_i$  do
6:       Learn consumer  $i$ 's taste  $\beta_{ik}$  using logit model with regularization
7:     end for
8:   end for
9:   Cluster all individual's tastes into  $K$  taste clusters
10:  for each consumer  $i = 1, \dots, I$  do
11:    for each individual product cluster  $k = 1, \dots, K_i$  do
12:      Assign products in  $k$  to the population taste which  $\beta_{ik}$  belongs to
13:    end for
14:  end for
15: end procedure

1: procedure ITERATIVESTEP
2:   repeat
3:     for each population taste cluster  $k = 1, \dots, K$  do
4:       Learn Mixed logit model with consumers and products that belong to this taste
5:       Get  $(\beta_k, \Sigma_k)$  and  $\beta_{ik}$ , i.e., individual posterior tastes for consumers who have taste  $k$ 
6:     end for
7:     for each consumer  $i$  do
8:       for each product  $j$  that consumer  $i$  purchased do
9:         Assign product  $j$  to taste  $k^*$ , where  $k^* = \arg \max_{k \in \mathbb{K}_i} \beta_{ik} x_j$ 
10:      end for
11:    end for
12:   until products taste cluster assignment for each consumer is the same as last iteration or
   objective function can not be decreased below tolerance  $\epsilon$ 
13: end procedure

```

to get a rough guess of individual tastes and also population tastes. For each consumer, we cluster his purchased products by product attributes, because the heterogeneity of the purchased products reveals the heterogeneity of tastes. As in the iterative-learning step, we also put a copy of products that were not purchased into each cluster. Then, for each product cluster, we can learn an individual taste using logistic regression with regularization. This procedure gives us an initialization of all the individual’s taste parameters. After obtaining all resulting individual’s tastes, we cluster them (as well as products that are associated with them) into K population taste clusters. This step gives us an initialization of population-level taste clusters.

We next test the performance of the IMML on simulated and empirical data.

5. Simulation

This section presents the results from a Monte Carlo experiment based on the heterogeneous multi-taste consumer model. We test whether the IMML algorithm can recover the true parameters for multi-taste consumers, and how it performs in terms of predicting held-out data compared with other choice models.

5.1. Data Generating Process and Simulations

In the Monte Carlo experiment, we have $K = 3$ tastes in the population, and each of them follows a normal distribution $N(b_i, \Sigma_k), k = 1, \dots, K$. We have $N = 100$ consumers and $J = 3000$ products with 30 binary attributes. The three taste parameters are shown in Table 1. To ensure the separate tastes are different from each other, we select two different attributes for each taste and put a mean weight of 4 on them. The variance for the weight is 0.1 for all attributes. For each consumer, we randomly select two tastes and draw the individual parameters for each of the tastes from the corresponding population distribution. Conditional on the individual parameters, we simulate each consumer’s choices on a random

	Taste 1	Taste 2	Taste 3
Attribute 1	4 (0.1)	0 (0.1)	0 (0.1)
Attribute 2	4 (0.1)	0 (0.1)	0 (0.1)
Attribute 3	0 (0.1)	4 (0.1)	0 (0.1)
Attribute 4	0 (0.1)	4 (0.1)	0 (0.1)
Attribute 5	0 (0.1)	0 (0.1)	4 (0.1)
Attribute 6	0 (0.1)	0 (0.1)	4 (0.1)
Attribute 7	0 (0.1)	0 (0.1)	0 (0.1)
Attribute 8	0 (0.1)	0 (0.1)	0 (0.1)
...
Attribute 30	0 (0.1)	0 (0.1)	0 (0.1)

Table 1 Simulation parameters (Taste parameters omitted are all with mean 0 and variance 0.1)

subset of products. We only simulate choices on 300 random products, rather than using all products, to capture the fact that in a real data set, a consumer is unlikely to have examined all the available products. The data are binary 1 or 0, 1 if the consumer has purchased the product, and 0 if the consumer did not purchase it. We generate 100 runs of the simulation.

5.2. Results

We estimate model parameters using the IMML algorithm, as well as a set of benchmark models.

Benchmark Models. We compared the results of the IMML algorithm with both single taste benchmark models and multi-taste benchmark models. Logit and mixed logit model are the single-taste benchmark models. They assume only one taste exists in the population, and hence we expect they will underfit. Latent class model assumes multiple taste segments in the population. It captures the market segments in the population; however,

they still assume each consumer belongs to only one segment. Cluster + Logit model clusters products by attributes first, and then learns consumers' tastes in each product cluster. In this case, each consumer could have multiple tastes. This approach is similar to the case in which we know how consumers categorize products and learn a taste per category. For example, when learning consumers' preference for movies, we can learn one taste per movie genre.

Parameter Recovery. Because in the simulation we know the true underlying parameter values that were used to generate the data, we can check whether a model can recover these parameters. Tables 6 and 7 in the Appendix summarize the parameter recovery. Table 6 shows the empirical mean and standard deviations of all the estimators based on 100 runs. Table 7 shows estimation results from one simulation run. For each estimator, we select the run that is the closest to the mean log-likelihood across the 100 runs. Unlike the other estimators, IMML always recovers the pattern of the taste distribution: it recovers three tastes with the correct attributes that have a high weight, and the other attributes are estimated to have weight 0. It is worth noting that, while the pattern of partworths is recovered, the exact values have bias in the constant and attribute partworths. This bias decreases when we run the same simulation for 1000 users. It takes about seven iterations on average to converge. As expected, both logit and mixed logit model underfit, and distribute utility weights evenly for the first six attributes. Latent class model performs much better than logit and mixed logit model, but each of their tastes is a combination of two correct tastes, because in the true model, each individual has two tastes. Hence $\binom{3}{2} = 3$ types of consumers exist in the population. Latent class model identifies these three types of customers but estimate each type of consumer as having only one taste, which is a combination of two tastes. As a result, they achieve the correct segmentation, but because

they do not capture the multi-taste disjunctive nature of the consumer preferences, they perform worse on out-of-sample prediction, which we discuss next.

Predictive Performance. Next, we test the predictive performance of IMML and compare it with that of the benchmark models. We randomly divide the data set into a training (80%) and a test (20%) set. We train IMML and all the benchmark models only in the training set, and test their predictive performance in test set. We compare all the models on five performance metrics: log-likelihood, accuracy, precision, recall, and F1. *Accuracy*, also called hit rate, is the percentage of predicted choices that match true choices. *Precision*, *recall*, and *F1* are widely used metrics in machine learning for classification and recommendation systems. Precision measures the percentage of predicted positives that are indeed true positives. In the context of choice modeling, it measures how many items that are predicted to be purchased actually are purchased. Recall (also called sensitivity in psychology) is the percentage of true positives that are correctly predicted to be positive. In our case, it measures how many items that are purchased were predicted to be purchased. F1 is the harmonic mean of precision and recall. These three measures focus on only the positive instances and predictions, and are widely used in the literature on recommender systems and information retrieval.

Table 2 shows the empirical mean and standard deviation of the predictive performance of all the estimators, averaged over the 100 simulation runs. We can see IMML outperforms the other benchmarks significantly on all the predictive metrics.

Convergence. On average over the 100 runs, IMML takes 6.9 iterations to converge. In each training iteration of IMML, products are reassigned to the taste clusters that give them the highest utility based on the parameter estimation from previous iterations. If the product taste assignment stays the same for two consecutive iterations, the algorithm converges. Figure 6 shows the number of products whose taste assignment changed decreases

	IMML	Logit	Cluster + Logit	LC	Mixed Logit
Log-likelihood	-2021.34*** (122.2)	-2587.95 (125.11)	-2567.36 (139.98)	-2275.42 (126.93)	-2426.14 (117.89)
Accuracy(Hit rate)	0.8947*** (0.0102)	0.8028 (0.0106)	0.8062 (0.0134)	0.8301 (0.0081)	0.8173 (0.0084)
Precision	0.86*** (0.0258)	0.6759 (0.0354)	0.6434 (0.0362)	0.7636 (0.0344)	0.6798 (0.0251)
Recall	0.6457*** (0.0307)	0.2647 (0.0367)	0.338 (0.0615)	0.3752 (0.0269)	0.4082 (0.0249)
F1	0.7278*** (0.0282)	0.379 (0.0389)	0.4408 (0.0573)	0.489 (0.0278)	0.4874 (0.024)

Table 2 Predictive performance across 100 runs of simulation studies. IMML significantly outperforms benchmark models on all metrics (*) $p < 0.001$.**

with each iteration, indicating that the algorithm stabilizes and converges to a saddle point. Figures 7 through 11 plot IMML’s out-of-sample predictive performance per each training iteration across 100 simulations for each of the five performance metrics. From these figures, we can see IMML improves with each iteration and becomes flat towards the end.

6. Empirical Application

This section presents an empirical application of the multi-taste consumer model and IMML estimation procedure in a real world setting. We select recipes as our focus category, because it exemplifies the type of category in which we expect multi-taste preferences to be relevant. Each ”product”, or recipe, is characterized by a large number of attributes, including ingredients and cooking methods (more detail on this below). Users select many recipes, giving them room for a diverse set of preferences. We collected data from a commercial recipe website called AllRecipes.com.¹ We apply our method to this data set and compare it with several popular alternative models.

¹ <http://allrecipes.com/>

6.1. The AllRecipes Data Set

Allrecipes.com is one of the largest food-focused online social network. This website’s users can post recipes, review others’ recipes, and collect recipes in their “recipe box”. We collected the data set in January 2015. The entire data set contains 136,012 users and 58,698 recipes. For each user, we collected recipes from their recipe box. For each recipe, we gathered the following information: a brief recipe introduction, ingredient list, cooking time, cooking directions, nutrition, and reviews and ratings. All of the collected information is in the form of unstructured text.

The website classifies recipes into categories, including “Appetizer”, “Breakfast and Brunch”, “Dessert”, etc. For our study, we focus only on recipes in the “Main Dish” and “Meat and Poultry” categories. These categories are diverse enough that we expect a single user might like different types of recipes and thus exhibit multi-taste behavior. At the same time, products in these two categories do not have an obvious sub-categorization. If we were to estimate the multi-taste model on recipes from all the categories, we would likely end up with degenerate tastes, e.g. a desert taste, a salad taste, a breakfast taste, etc. This is not what we want. By focusing only on main dish and meat and poultry recipes, we are able to discover more surprising and subtle underlying patterns in the population preferences. In total, we have 121,208 users who selected at least one recipe in either the “Main dish” or the “Meat and Poultry” categories, and a total of 12,775 recipes. On average, each user had about 42 recipes in their recipe box.

We expect that multi-taste preferences are more likely to be observed for users with large recipe boxes. In addition, a challenge we face in this application is the sparsity of the data: most users do not select most recipes. In fact, the average recipe on the site is only present in 0.3% users’ recipe boxes, and 95% of recipes are present in less than 1.4% recipe boxes.

For these reasons, we focus our analysis on popular recipes and active users. We sample 1,000 recipes and 1,000 users, creating a data set with 1 million observations, as follows. The 1,000 recipes in the data used for estimation are randomly selected from the 2,000 most popular recipes (the remaining 1,000 are used for feature selection, as described in section 6.2). Popular recipes are those most frequently occurring in recipe boxes. Selecting the top 2000 recipes results in a cutoff of occurring in at least 621 recipe boxes (out of the total 121,208 users). We then randomly select 1,000 users who have at least 100 of the sampled recipes in their recipe box. In addition, we removed the top one percentile of active users - many of these selected almost every recipe, and may be robots. The resulting data set is still very sparse compared to most choice modelling settings, with about 88% negative observations ².

6.2. Product Attributes Extraction and Selection

The first step of modeling consumer choices is to extract important product attributes from unstructured text that we collected. Of the information available in the recipe text, we use two types of information as product attributes: (1) ingredients, e.g., “egg”, “bean”, and (2) cooking methods, e.g., “roast”, “saute”.

We automatically extract ingredients from the ingredient list portion of the recipe. An example of an ingredient list item as it occurs in the raw data is “1 cup shredded cheddar cheese, divided.” We use a part-of-speech (POS) tagger to annotate each word in the recipe ingredient list to identify whether a word is a number, an adjective, a noun, or a verb. Then we use regular expression to remove non-ingredient terms from the list, specifically numbers, modifiers or verb. We extract a total of 1,077 ingredients. We identify seven cooking methods using a listing in the Wikipedia entry on cooking as used in Teng et al.

²For robustness, we also constructed a different, more random sample of users and recipes and estimated all the models on it. The results is similar

(2012): bake, boil, fry, grill, roast, simmer, and marinate. We extract whether each recipe uses these cooking methods by mining the cooking directions portion of the recipe.

Next, we do feature selection using ℓ_1 -regularization techniques. We estimate a lasso estimator based on the observations of the recipes hold out for feature selection (Tibshirani (1996)). It minimizes the negative log-likelihood along with an ℓ_1 -penalty:

$$\hat{\beta}^{lasso} = \arg \min_{\beta} \ell(\{\beta; y, X\}) + \lambda \|\beta\|_1. \quad (11)$$

The ℓ_1 penalty causes some coefficients to be set to exactly zero which allows us to do feature selection. The λ parameter controls the strength of the penalty, and hence the sparsity of the model. As λ increases, fewer features are selected. λ is set through cross-validation. This procedure results in 84 attributes, including three cooking methods and 81 ingredients.

The resulting data set consists of 1 million observations of 1000 consumers' choices on 1000 recipes. Each data point is a binary choice on whether or not to collect the recipe in the recipe box. Each recipe is represented using 84 binary attributes, where each attribute indicates whether or not the recipe contains the ingredient or cooking method. We randomly divide the data into two sets: 80% of the data is used for training and calibrating the model and 20% of the data is held out for performance comparison with other models. When dividing the data, for each user, we randomly select as training data 80% of the recipes he selected and 80% of the recipes he didn't select, and hold out the rest as test data. This way of dividing data into training and test has two advantages over simply holding out a set of recipes for all users. First, the distribution of choices for each individual is similar in training data and in hold out data. Second, it more closely mimics the real-world setting, in which we do not always have observations of the same set of products for all consumers.

6.3. Model Estimation and Results

On the resulting training set, we apply the IMML procedure to estimate the multi-taste preference model. We compare its predictive performance with four benchmark models. We call the first benchmark model a “Null” model. It simply predicts all negatives. The second and third model are single-taste benchmarks: Logit (homogeneous consumers) and Mixed logit model (heterogeneous consumers). The fourth benchmark, which we call “Cluster + Logit” also captures multi-taste preferences. It first performs a K -means clustering on all the recipes’ attributes to group recipes to different types. Number of clusters is determined using the silhouette method (Rousseeuw (1987)). Then it fits a model that includes a full set of interactions of recipe types (clusters) and recipe attributes.

$$U_j = \alpha + \sum_{a=1}^A \beta_a x_{ja} + \sum_{k=2}^K \gamma_k s_{jk} + \sum_{k=2}^K \sum_{a=1}^A \theta_{ka} (s_{jk} \times x_{ja}) + \varepsilon_j, j = 1, \dots, J \quad (12)$$

where x_{ja} is attribute a of recipe j , s_{jk} is the dummy variable for whether recipe j belongs to cluster k . In this benchmark model, each consumer has the same number of tastes as the number of recipe clusters. The resulting number of parameters to be estimated is $K * (A + 1)$, where K is number of recipe clusters and A is number of recipe attributes.

Table 3 gives the out-of-sample prediction performance of our model the above four benchmark models. For our model, we fit the model with K number of tastes in population for $K = 2, \dots, 6$.

We compare the models on a set of six performance metrics: log-likelihood, accuracy, precision, recall, F1, and number of true positives predicted. All the metrics are computed based on prediction results on the held-out data. First, note that due to the sparsity of the data, with the majority of data being comprised of negative observations, the null model achieves an accuracy of 88% by predicting all negative. However, this is clearly not a very useful model, because it does not help us understand population preferences or make any

	Log-likelihood	Accuracy	Precision	Recall	F1	True Positives
Null	-72742	88.24%	NaN	0.00%	0.00%	0
Logit	-71789	88.24%	NaN	0.00%	0.00%	0
Mixed Logit	-71179	88.24%	66.67%	0.01%	0.02%	2
Cluster + Logit	-70617	88.24%	NaN	0.00%	0.00%	0
IMML (K=2)	-71050	88.25%	55.79%	0.23%	0.45%	53
IMML (K=3)	-71796	88.21%	39.86%	0.49%	0.97%	116
IMML (K=4)	-71696	88.21%	39.65%	0.57%	1.12%	134
IMML (K=5)	-71982	88.22%	42.91%	0.49%	0.97%	115
IMML (K=6)	-72123	88.23%	43.78%	0.43%	0.86%	102

Table 3 Empirical application: hold out sample predictive performance

predictions about which users might like which recipes. We also note that the log-likelihood is quite flat among the tested models. The model that has the highest log likelihood is the Cluster + Logit multi-taste model. This model, however, also does not predict any positives. We introduce the precision, recall, and F1 measures, common in recommender systems literature. We can see that the IMML-based multi-taste models are able to predict positives, even though the resulting log-likelihood is slightly lower than the Mixed Logit and Cluster + Logit benchmarks. They also significantly outperform the benchmarks on the F1 measure. By comparing the IMML models with $K = 2, \dots, 6$ on the F1 measure, which balances precision and recall, and the true positives measure, IMML model with 4 tastes in population seems to perform the best. Next, we look at the estimates of this model and characterize the underlying population preference parameters.

Table 4 IMML Taste Estimates (K = 4)

IMML					
	Taste 1 <i>High effort</i>	Taste 2 <i>Italian/pasta</i>	Taste 3 <i>Chicken/healthy</i>	Taste 4 <i>Trad. American</i>	Mixed Logit
Top	butter	ground_beef	curd_cottage_cheese	cracker	curd_cottage_cheese
	lime_juice	pasta	tomato_puree	dill_pickles	tomato_puree
	curd_cottage_cheese	ricotta_cheese	cracker	roast	lime_juice
	clove	tomato_puree	lime_juice	tomato_puree	roast
	mustard	simmer	garlic_salt	soy_sauce	soy_sauce
	mustard_powder	mustard_powder	soy_sauce	Cajun_seasoning	mustard_powder
	chicken_breast_half	pea	onion_flake	mushroom_soup	pasta
	bread_dough	lasagna_noodle	chicken_breast	onion_flake	garlic_salt
	leaf	roast	pepper_flake	curd_cottage_cheese	ricotta_cheese
	Dijon_mustard	Cheddar_cheese	chicken_breast_half	beef_brisket	chicken_breast_half
Bottom	chicken_soup	clove	curry_powder	seed	dog_bun
	soy_sauce	tarragon	rib	dog_bun	bread_dough
	brandy	iron_steak	chicken_soup	sesame_oil	ground_beef_chuck
	pizza_sauce	jasmine_rice	firm_tofu	tarragon	beef_sirloin
	mayonnaise	saffron_thread	saffron_thread	ground_beef_chuck	rib
	sherry	chipotle_pepper	jasmine_rice	ricotta_cheese	chinese_food
	dog_bun	kernel_corn	thyme_leaf	simmer	jasmine_rice
	beef_brisket	cracker	cheese_food	ground_beef	firm_tofu
	pasta	onion_flake	ground_beef	firm_tofu	thyme_leaf
	cider_vinegar	butter	butter	butter	saffron_thread

Table 4 shows the top 10 attributes and bottom 10 attributes for both multi-taste model (IMML) and single taste model (Mixed Logit), for comparison. For each taste, we rank recipe attributes by their coefficients, from highest (most positive) to lowest (most negative). Table 8 in the appendix shows the coefficient estimates of the full set of recipe attributes.

The resulting categorization of the multi-taste estimates can be interpreted as follows. The first taste involves **High-effort recipes made from scratch**. It has a high weight on “butter,” “lime juice,” “cottage cheese,” “clove,” “bread dough,” meats (“chicken breast half,” “ground beef”), and vegetables (“leaf,” “corn”), and negative weights on ingredients such as “pizza sauce,” “ranch dressing,” “chicken soup,” and “spaghetti sauce.”

The second taste is **Italian/pasta dishes**. The top ingredients are “pasta,” “ground beef,” “tomato puree,” and “ricotta cheese.” Negative weight is placed on, for example, “butter,” “jasmine rice,” “iron steak,” “avocado,” and “chipotle peppers.”

The third taste suggests a preference for **Chicken and healthy/low-calorie cooking**. It has high weight on “cottage cheese,” “tomato puree,” low-fat meat (“chicken breast half,” “chicken breast”), and seasonings (“soy sauce,” “lime juice,” “garlic salt,” “pepper flake,” and “zest”). It has negative weight on high fat-content ingredients, such as “butter,” “cheese food,” “cheese ravioli,” and meats (“ground beef,” “ground beef chuck,” “beef sirloin,” and “rib”).

The fourth taste is **Traditional American**: high weight on “tomato puree,” “roast,” “cracker,” “mushroom soup,” “barbecue sauce,” and “beef brisket,” and negative weights on “butter,” “ricotta cheese,” “simmer,” “avocado,” “jasmine rice,” “sesame oil,” and “ginger root.”

The resulting estimates from the single taste model can be interpreted as a combination of all the tastes above, particularly the chicken taste and pasta tastes: high weight on

Table 5 Butter Estimates

IMML					Mixed Logit
Taste 1 <i>High effort</i>	Taste 2 <i>Italian/pasta</i>	Taste 3 <i>Chicken/healthy</i>	Taste 4 <i>Traditional American</i>		
6.17	-4.62	-7.15	-10.46		-0.0094

“lime juice,” “cottage cheese,” “pasta,” “ricotta cheese,” “ground beef,” “chicken breast half.” Estimating a single taste rather than multiple tastes also results in estimates of some coefficients to be close to 0, even though the multi-taste model reveals that the same attribute has non-zero coefficients in different tastes. For example, consider the coefficient of the attribute “butter”, presented in Table 5. In the mixed logit model, it is estimated to have a value of 0.0064, while ranging from -10.46 to 6.16 in the multi-taste model, depending on the taste.

6.3.1. Taste Distribution Across Consumers Because the key aspect of our model relative to existing choice models is that each consumer can have more than one taste, and consumers can have different subsets of tastes, it allows us to look at the distribution of tastes among consumers. Figure 2 displays the histogram of the number of tastes per consumer: the majority of consumers have two tastes and three tastes. Each consumer has an average of 2.43 tastes. Figure 3 shows number of consumers who has each taste. The first three tastes are popular tastes with a lot of consumers but only 30% of consumers have the last taste.

Given that we know the subset of tastes that each consumer has, we can also compute number of consumers with a particular subset of tastes to understand how tastes co-occur. Figure 6 and Figure 7 display the number of consumers with a particular pair of tastes and each set of three of tastes, respectively. From the figures we can see that the third taste (chicken/healthy) and the fourth taste (traditional american) co-occur relatively rarely -

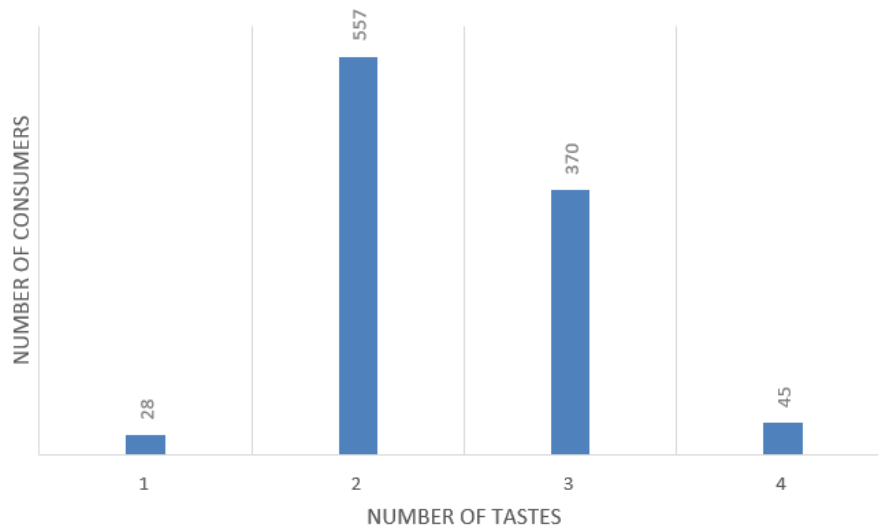


Figure 2 Number of tastes per consumer.

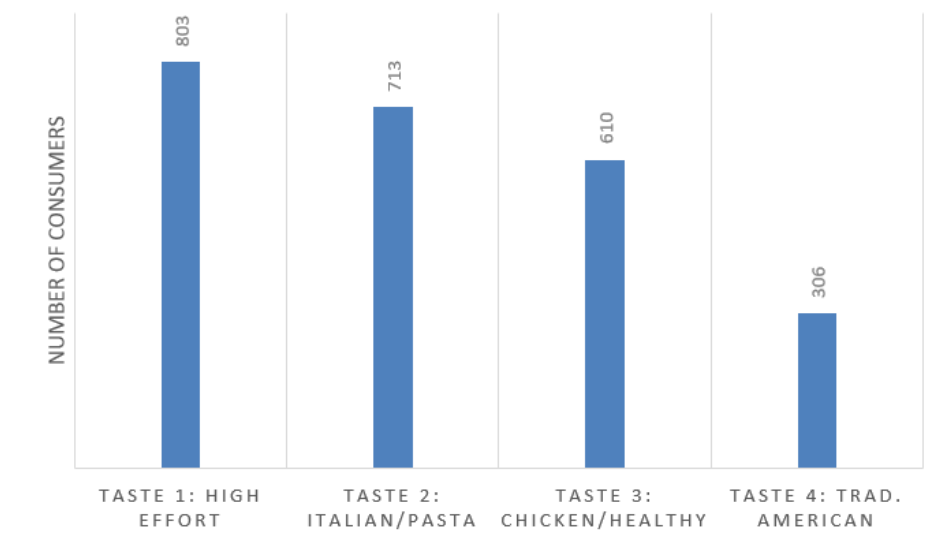


Figure 3 Number of consumers with each taste.

only 4% of consumers have both of these tastes, suggesting that very few consumers cook both these types of recipes. However, 20% of consumers have both the healthy/chicken and the high effort taste. The flexibility provided by our model and estimation procedure allows us to find these complex patterns in population preferences.

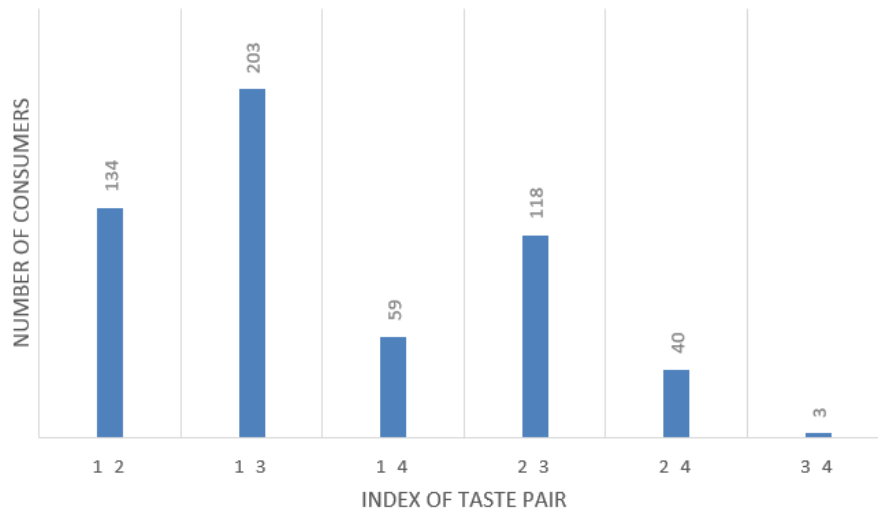


Figure 4 Number of people with each pair of tastes.

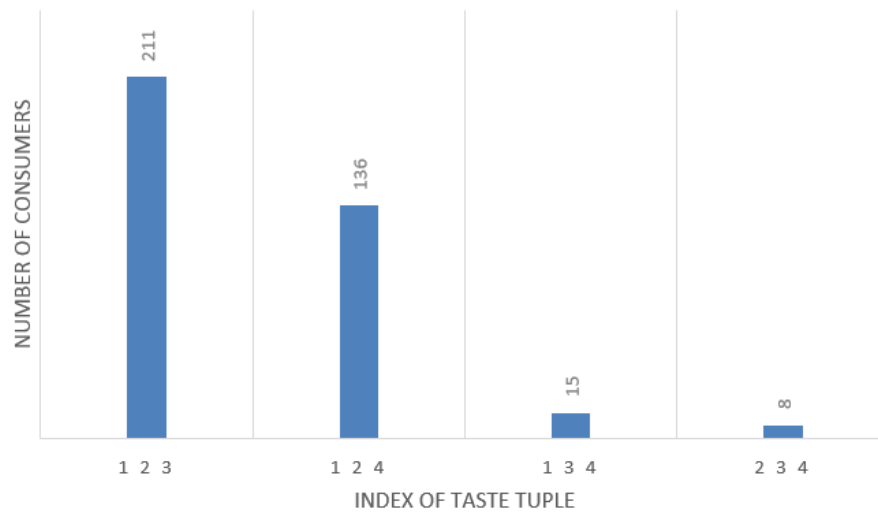


Figure 5 Number of consumers with each tuple of tastes.

7. Conclusion

We present a new approach for modeling and estimating consumer preferences from purchase data that is well tailored for diverse product categories, in which one consumer may have multiple different tastes. Our analysis has demonstrated that:

1. The proposed algorithm is able to successfully recover the true parameters of a multi-taste model in simulations. It is substantially more effective than existing benchmarks.
2. The method scales well to a large, high-dimensional data set (84 attributes, 1000 products) and makes better predictions on hold-out data than a single-taste model.
3. The majority of consumers exhibit multi-taste behavior, with most consumers (about 56%) being fit best with a 2 taste model, and about 37% fit best with a three taste model.
4. Accounting for multiple tastes of a single consumer also generates a meaningful segmentation of consumer preferences.

The first finding demonstrates the algorithm performs well in simulation and succeeds at recovering parameters. This is critical because the IMML is a heuristic algorithm and is not theoretically guaranteed to converge to the correct estimate. The simulation studies allow us to test its ability to recover the correct parameters, not only make accurate predictions. The second result demonstrates its ability to perform well in practice, on large problems and further demonstrates predictive performance. Finally, the last two results suggest important substantive insights can be gained from the proposed method, and shows evidence that consumer do indeed use multiple tastes in a diverse product category like recipes.

Appendix

A. CCCP Algorithm for Homogeneous Multi-taste Consumers

Algorithm 2 The CCCP Algorithm for Homogeneous Multi-taste Consumers

Input: $\mathcal{D} = \{(x_1, y_1), \dots, (x_J, y_J)\}$ // Consumer's choice on all J products

K // Number of tastes

ϵ // Error threshold

Output: $\mathcal{B} = \{\beta_1, \dots, \beta_K\}$

1: $t \leftarrow 0$ and initialize $\mathcal{B}^{(0)}$

2: **repeat**

3: $k_j^* = \arg \max_k \beta_k^T x_j$

4: Update $\mathcal{B}^{(t+1)}$ by fixing k_j^* for (x_j, y_j) and minimizing the resulting convex sum, specifically

$$\mathcal{B}^{(t+1)} = \arg \min_{\{\beta_1, \beta_2, \dots, \beta_K\}} \left[\sum_{j=1}^n \max_k \log(1 + e^{\beta_k x_j}) - \sum_{j=1}^n y_j \beta_{k_j^*}^T x_j \right]$$

5: **until** Objective function cannot be decreased below tolerance ϵ

B. EM for Heterogeneous Multi-taste Consumer Model

The EM algorithm is derived following notations and steps in Train (2008). Our goal is to estimate $\theta = \{b_1, \Sigma_1, \dots, b_K, \Sigma_K, \alpha_{i1}, \dots, \alpha_{iK}\}$ by solving the maximum likelihood function

$$\arg \max_{\theta} \sum_i \log P_i(\theta)$$

There exists two types of missing variable: (1) tastes membership \mathbb{K}_i , i.e., whether consumer i has taste subset \mathbb{K}_i ; (2) $\{\beta_{ik}, \forall k \in \mathbb{K}_i\}$ for each consumer i , i.e., individual parameters. Conditional on the customer's choices, the probability-density of $\{\beta_{ik}, k \in \mathbb{K}_i\}$ and tastes subset \mathbb{K}_i is

$$h_{i\mathbb{K}_i}(\{\beta_{ik}, \forall k \in \mathbb{K}_i\}|\theta) = \frac{\left(s_{\mathbb{K}_i} \prod_{k \in \mathbb{K}_i} \phi(\beta_{ik}|b_k, \Sigma_k)\right) Q_i(\{\beta_{ik}, \forall k \in \mathbb{K}_i\})}{P_i(\theta)} \quad (13)$$

where $s_{\mathbb{K}_i}$ (Equation (8)), $Q_i(\{\beta_{ik}, \forall k \in \mathbb{K}_i\})$ (Equation (5)), and $P_i(\theta)$ (Equation (7)) is defined in main text.

The expectation in the EM algorithm is

$$\begin{aligned} \mathcal{E}(\theta|\theta^t) &= \sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \int \cdots \int h_{i\mathbb{K}_i}(\{\beta_{ik}, \forall k \in \mathbb{K}_i\}|\theta^t) \\ &\quad \log \left[s_{\mathbb{K}_i} Q_i(\{\beta_{ik}, \forall k \in \mathbb{K}_i\}) \prod_{k \in \mathbb{K}_i} \phi(\beta_{ik}|b_k, \Sigma_k) \right] d\beta_{ik_1} \cdots d\beta_{ik_{|\mathbb{K}_i|}} \end{aligned} \quad (14)$$

Substituting (13) and rearranging gives

$$\begin{aligned} \mathcal{E}(\theta|\theta^t) &= \sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \int \cdots \int \left[s_{\mathbb{K}_i} Q_i(\{\beta_{ik}, \forall k \in \mathbb{K}_i\}) / P_i(\theta) \right] \\ &\quad \log \left[s_{\mathbb{K}_i} Q_i(\{\beta_{ik}, \forall k \in \mathbb{K}_i\}) \prod_{k \in \mathbb{K}_i} \phi(\beta_{ik}|b_k, \Sigma_k) \right] \prod_{k \in \mathbb{K}_i} \phi(\beta_{ik}|b_k, \Sigma_k) d\beta_{ik_1} \cdots d\beta_{ik_{|\mathbb{K}_i|}} \end{aligned} \quad (15)$$

Both the expectation (Equation (15)) and $P_i(\theta)$ involves integrals over $\phi(\cdot)$. This integration can be approximated by simulation by taking R draws from each normal distribution of the tastes in consumer i 's taste subset. Let β_{ikr} for draw r from normal $N(b_k, \Sigma_k)$ for consumer i . We can write simulated choice probability as

$$\tilde{P}_i(\theta) = \sum_{\mathbb{K}_i \subseteq \mathbb{K}} s_{\mathbb{K}_i} \sum_r Q_i(\{\beta_{ikr}, \forall k \in \mathbb{K}_i\}) / R \quad (16)$$

and simulated expectation as

$$\tilde{\mathcal{E}}(\theta|\theta^t) = \sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \sum_r h_{i\mathbb{K}_i}^t \log \left[s_{\mathbb{K}_i} Q_i(\{\beta_{ikr}, k \in \mathbb{K}_i\}) \prod_{k \in \mathbb{K}_i} \phi(\beta_{ikr}|b_k, \Sigma_k) \right] / R \quad (17)$$

where $h_{i\mathbb{K}_i}^t = s_{\mathbb{K}_i} Q_i(\{\beta_{ikr}, k \in \mathbb{K}_i\}) / \tilde{P}_i(\theta^t)$

In the maximization step, we solve

$$\theta^{t+1} = \arg \max_{\theta} \tilde{\mathcal{E}}(\theta|\theta^t) \quad (18)$$

Notice that the complete data log likelihood part in Equation (15) can be written as the sum of three parts

$$\begin{aligned} & \log \left[s_{\mathbb{K}_i} Q_i(\{\beta_{ikr}, k \in \mathbb{K}_i\}) \prod_{k \in \mathbb{K}_i} \phi(\beta_{ikr} | b_k, \Sigma_k) \right] \\ &= \log s_{\mathbb{K}_i} + \log \prod_{k \in \mathbb{K}_i} \phi(\beta_{ikr} | b_k, \Sigma_k) + \log Q_i(\{\beta_{ikr}, k \in \mathbb{K}_i\}). \end{aligned}$$

We can optimize separately for each set of parameters. Also notice that $Q_i(\{\beta_{ikr}, k \in \mathbb{K}_i\})$ does not depend on the parameters. It depends on the random draws of β_{ikr} . Once the draws are given, change of parameters of θ will only affect $\phi(\beta_{ikr} | b_k, \Sigma_k)$ but not $Q_i(\{\beta_{ikr}, k \in \mathbb{K}_i\})$, and hence we can drop $Q_i(\{\beta_{ikr}, k \in \mathbb{K}_i\})$ from optimization function. The resulting maximization becomes:

$$\{\alpha_1, \dots, \alpha_K\}^{t+1} = \arg \max_{\{\alpha_1, \dots, \alpha_K\}} \sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \sum_r h_{i\mathbb{K}_i r}^t \log s_{\mathbb{K}_i}, \quad (19)$$

and

$$\{b_1, \Sigma_1, \dots, b_K, \Sigma_K\}^{t+1} = \arg \max_{\{b_1, \Sigma_1, \dots, b_K, \Sigma_K\}} \sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \sum_r h_{i\mathbb{K}_i r}^t \log \prod_{k \in \mathbb{K}_i} \phi(\beta_{ikr} | b_k, \Sigma_k). \quad (20)$$

The maximization in Equation (19) is satisfied by

$$\alpha_k^{t+1} = \frac{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}: k \in \mathbb{K}_i} \sum_r h_{i\mathbb{K}_i r}^t}{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \sum_r h_{i\mathbb{K}_i r}^t}, \quad k = 1, \dots, K. \quad (21)$$

The maximization in Equation (20) is satisfied by

$$b_k^{t+1} = \frac{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}: k \in \mathbb{K}_i} \sum_r h_{i\mathbb{K}_i r}^t \beta_{ikr}}{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \sum_r h_{i\mathbb{K}_i r}^t}, \quad k = 1, \dots, K, \quad (22)$$

and

$$\Sigma_k^{t+1} = \frac{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}: k \in \mathbb{K}_i} \sum_r h_{i\mathbb{K}_i r}^t \left[(\beta_{ikr} - b_k^{t+1}) (\beta_{ikr} - b_k^{t+1})' \right]}{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \sum_r h_{i\mathbb{K}_i r}^t}, \quad k = 1, \dots, K. \quad (23)$$

We list the pseudocode of this EM algorithm in Algorithm 3.

Algorithm 3 The EM Algorithm for Heterogeneous Multi-taste Consumers

Input: $\mathcal{D} = \{(x_1, y_1), \dots, (x_J, y_J)\}$ // Consumer's choice on all J products
 K // Number of tastes
 ϵ // Error threshold

Output: $\theta = \{b_1, \Sigma_1, \dots, b_K, \Sigma_K, \alpha_{i1}, \dots, \alpha_{iK}\}$

- 1: $t \leftarrow 0$ and initialize θ^0 .
- 2: **repeat**
- 3: **for** each consumer $i = 1, \dots, I$ **do**
- 4: **for** each normal $k = 1, \dots, K$ **do**
- 5: Take R draws from k -th normal $N(b_k^t, \Sigma_k^t)$. Label r -th draw as β_{ikr}
- 6: **end for**
- 7: **for** each possible taste subset, i.e., $\forall \mathbb{K}_i \in \mathbb{K}$ **do**
- 8: **for** each draw $r = 1, \dots, R$ **do**
- 9: Compute $h_{i\mathbb{K}_i r}^t = s_{\mathbb{K}_i} Q_i(\{\beta_{ikr}, k \in \mathbb{K}_i\}) / \tilde{P}_i(\theta^t)$
- 10: **end for**
- 11: **end for**
- 12: **end for**
- 13: Update

$$\alpha_k^{t+1} = \frac{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}: k \in \mathbb{K}_i} \sum_r h_{i\mathbb{K}_i r}^t}{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \sum_r h_{i\mathbb{K}_i r}^t}, \quad k = 1, \dots, K$$
- 14: Update

$$b_k^{t+1} = \frac{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}: k \in \mathbb{K}_i} \sum_r h_{i\mathbb{K}_i r}^t \beta_{ikr}}{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \sum_r h_{i\mathbb{K}_i r}^t}, \quad k = 1, \dots, K$$
- 15: Update

$$\Sigma_k^{t+1} = \frac{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}: k \in \mathbb{K}_i} \sum_r h_{i\mathbb{K}_i r}^t \left[(\beta_{ikr} - b_k^{t+1}) (\beta_{ikr} - b_k^{t+1})' \right]}{\sum_i \sum_{\mathbb{K}_i \subseteq \mathbb{K}} \sum_r h_{i\mathbb{K}_i r}^t}, \quad k = 1, \dots, K$$
- 16: $t \leftarrow t + 1$
- 17: **until** Objective function cannot be decreased below tolerance ϵ

C. Tables

<i>Attributes</i>	True tastes			Logit	MIXL	Cluster + Logit			LC			IMML		
	β_1	β_2	β_3			β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3
Intercept	-6	-6	-6	-3.47	-3.62	-2.26	-3.88	-3.86	-4.34	-4.07	-4.17	-7.49	-7.73	-7.73
Attribute 1	4	0	0	1.11	1.31	0.75	1.50	1.13	1.95	-0.02	1.77	4.86	-0.40	-0.05
Attribute 2	4	0	0	1.16	1.32	0.80	1.30	1.06	1.95	0.05	1.92	5.06	-0.38	-0.08
Attribute 3	0	4	0	1.06	0.59	2.29	0.45	1.19	1.85	1.89	-0.07	-0.33	5.51	-0.33
Attribute 4	0	4	0	1.07	NA	NA	NA	1.26	1.77	1.81	0.09	-0.16	5.51	-0.28
Attribute 5	0	0	4	1.05	1.25	0.80	1.37	1.43	0.17	1.93	1.83	-0.04	-0.10	5.15
Attribute 6	0	0	4	1.06	1.46	0.80	1.48	1.32	0.01	1.99	2.03	-0.11	-0.54	5.45

Table 6 Parameter estimates from a typical simulation run. For each estimator, we select the run that is closest to the mean log-likelihood across 100 runs. IMML recovers parameters best.

Attributes	True tastes			Logit	MIXL	Cluster + Logit			LC			IMML		
	β_1	β_2	β_3			β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3
Intercept	-6	-6	-6	-3.61 (0.09)	-3.9 (0.1)	-3.53 (0.45)	-3.47 (0.5)	-3.54 (0.44)	-4.13 (0.33)	-4.11 (0.18)	-4.07 (0.19)	-8.48 (4.46)	-8.67 (6.62)	-8.31 (3.91)
Attribute 1	4	0	0	1.12 (0.09)	1.22 (0.12)	1.08 (0.26)	1.04 (0.26)	1.3 (0.36)	1.88 (0.19)	0.05 (0.3)	1.84 (0.28)	5.11 (1.34)	-0.03 (0.79)	-0.06 (0.91)
Attribute 2	4	0	0	1.11 (0.08)	1.18 (0.11)	1.07 (0.19)	1.09 (0.27)	1.24 (0.26)	1.84 (0.32)	0.03 (0.27)	1.84 (0.27)	5.11 (1.28)	-0.08 (0.68)	-0.07 (0.93)
Attribute 3	0	4	0	1.11 (0.08)	1.17 (0.13)	1.06 (0.21)	1.24 (0.34)	1.1 (0.27)	1.86 (0.24)	1.89 (0.18)	0.06 (0.29)	0.19 (2.78)	5.38 (2.16)	-0.15 (0.53)
Attribute 4	0	4	0	1.1 (0.08)	1.19 (0.11)	1.07 (0.2)	1.23 (0.34)	1.08 (0.25)	1.86 (0.26)	1.87 (0.18)	0.07 (0.33)	0.23 (2.76)	5.39 (2.31)	-0.17 (0.61)
Attribute 5	0	0	4	1.12 (0.1)	1.19 (0.13)	1.25 (0.27)	0.94 (1.58)	1.09 (0.26)	0.07 (0.36)	1.86 (0.22)	1.82 (0.24)	-0.04 (1.5)	0.09 (2.03)	5.3 (1.59)
Attribute 6	0	0	4	1.13 (0.09)	1.21 (0.13)	1.32 (0.34)	1.09 (0.37)	1.02 (0.26)	0.08 (0.36)	1.86 (0.26)	1.84 (0.24)	-0.03 (1.42)	0.07 (1.97)	5.31 (1.59)

Table 7 Empirical means and standard deviations of estimators from 100 runs of simulations. IMML recovers parameters best.

	Taste 1	Taste 2	Taste 3	Taste 4	Mixed Logit
	High effort	Italian/pasta	Chicken/healthy	Trad. America	
constant	-8.3097	-5.1271	-2.3129	-2.0281	-2.1530
butter	6.1673	-4.6189	-7.1497	-10.4628	-0.0094
chicken_breast_half	0.4772	0.0809	0.3082	-0.1006	0.2377
tomato	-0.4980	0.1042	0.0565	-0.4822	-0.0106
Parmesan_cheese	0.0157	0.2990	0.2475	-0.0128	0.0762
soy_sauce	-0.6071	-0.0275	0.3623	0.4692	0.2822
parsley	0.2360	-0.2170	-0.0135	0.1043	0.1196
ground_beef	0.2818	2.6487	-2.6506	-3.0919	0.2010
all-purpose_flour	-0.5162	0.1040	0.0675	0.0088	-0.1030
Cheddar_cheese	-0.2745	0.4166	0.0608	-0.3457	-0.0331
cayenne_pepper	-0.1019	-0.4420	0.0518	-0.0701	-0.0315
mustard	0.7593	0.1640	-0.2795	-0.3287	0.0484
ground_cumin	-0.0231	0.1248	0.0613	-1.2237	-0.1373
mayonnaise	-0.7665	-0.1133	0.2076	-0.0609	0.0257
pepper_flake	-0.2543	0.3068	0.3100	-0.2748	0.2303
mushroom_soup	-0.0033	0.0099	-0.1587	0.3721	0.0485
rosemary	-0.0726	-0.3212	-0.1062	-0.2223	-0.1703
flour	-0.1303	-0.1584	-0.3148	-0.5994	-0.2442
rice	-0.3900	0.1475	-0.1293	-0.6216	-0.1875
tomato_sauce	-0.1373	-0.0935	0.2259	-0.3435	0.0605
salsa	-0.0956	0.3757	0.1364	-0.7831	0.0869
chicken_breast	0.1924	0.2368	0.3305	-0.4967	0.1553
chicken_soup	-0.5297	-0.0625	-0.7386	-0.2786	-0.4021
Dijon_mustard	0.3931	-0.0682	0.1794	-0.0119	0.1936
spaghetti_sauce	-0.3168	-0.0036	-0.3646	-0.5698	-0.2210
ginger_root	-0.0635	-0.1884	-0.1668	-1.0693	-0.3087
leaf	0.3979	-0.4013	-0.1785	0.1667	-0.0707
ricotta_cheese	-0.0797	1.5326	-0.1985	-1.6143	0.2518
lime_juice	1.8096	0.0820	0.3981	-0.2627	0.4924
barbeque_sauce	-0.3057	-0.1271	0.0133	0.2677	0.0857
curry_powder	-0.1477	-0.5051	-0.5106	-0.9784	-0.4929
lasagna_noodle	-0.3116	0.5223	-0.4945	-0.6479	0.0789
sesame_oil	-0.2509	-0.1040	-0.1531	-1.3010	-0.2859
garlic_salt	0.0003	0.2239	0.3669	0.0999	0.2524
pea	-0.0689	0.7010	0.2182	0.1702	0.1417
salmon_fillet	0.0834	-0.2060	-0.1103	-0.2772	-0.3050
roast	0.1655	0.4278	0.2750	0.5787	0.3172
avocado	-0.1134	-0.5496	-0.4029	-1.1167	-0.6142
mustard_powder	0.6239	0.8321	-0.1474	-0.1357	0.2659
clove	0.8027	-0.5942	-0.2880	0.0127	0.1215
Cajun_seasoning	0.3764	0.0514	0.0450	0.4201	0.1555
angel_hair_pasta	0.1674	0.1246	0.0717	-0.2819	0.0720
celery_salt	-0.1910	-0.1431	-0.2145	-0.9175	-0.4379
asparagus	0.1646	-0.3069	-0.0209	-0.4102	-0.1635
kernel_corn	0.2611	-0.8803	-0.3930	-0.7114	-0.5688
zest	-0.1756	-0.0644	0.2659	-0.5097	0.0673
pie_crust	0.2015	0.0021	0.1380	-0.2748	0.0837

sherry	-0.8065	-0.1970	-0.3165	-0.3230	-0.3960
cider_vinegar	-1.2055	-0.2968	0.1531	0.0768	-0.2247
hoagie_roll	-0.2795	-0.4300	0.0821	0.1522	0.0069
beef_brisket	-0.9203	0.0689	0.1185	0.3311	-0.0185
pizza_sauce	-0.7312	-0.0989	-0.1609	-0.5296	-0.4242
seed	0.1223	-0.5021	-0.3806	-1.2430	-0.6382
tarragon	-0.2126	-0.6071	-0.3095	-1.4230	-0.3728
firm_tofu	0.1187	-0.4879	-0.9485	-3.4376	-1.1772
chipotle_pepper	-0.0332	-0.8332	-0.0052	-0.2244	-0.2815
cilantro_leaf	-0.0086	-0.1435	0.2269	-0.6948	-0.0427
pasta	-1.0703	2.0840	-0.4470	-0.2423	0.2651
cracker	0.1043	-0.9873	0.4115	1.0451	-0.1053
thyme_leaf	-0.0809	-0.0500	-1.1960	-0.7476	-1.1960
beef_sirloin	0.0012	-0.1999	-0.4092	-1.1573	-0.8660
iceberg_lettuce	0.3310	0.1315	-0.0982	-0.9816	-0.0937
cheese_food	-0.2653	-0.1077	-1.4251	-0.7859	-0.9321
cheese_ravioli	-0.0412	0.3058	-0.5085	-0.7233	-0.2442
bread_dough	0.4003	-0.0870	-0.0907	0.0033	-0.6828
ranch_dressing	-0.3667	0.1774	0.0547	-0.4222	-0.2649
rib	-0.0111	-0.2343	-0.7352	-1.0575	-0.8791
beef_sirloin_steak	0.0691	-0.0719	-0.1668	-0.7637	-0.1815
iron_steak	-0.1836	-0.6809	-0.1881	-0.2444	-0.4630
dill_pickles	0.0760	0.0510	0.1272	0.7576	0.1301
brandy	-0.6280	0.0651	-0.4983	-0.6029	-0.5039
jasmine_rice	-0.2712	-0.7560	-1.0573	-0.9303	-1.0810
dog_bun	-0.8102	-0.0163	-0.2302	-1.2527	-0.6401
curd_cottage_cheese	1.2028	0.0814	1.0161	0.3582	1.0834
tomato_puree	-0.1379	1.0794	0.7670	0.5121	0.6481
chorizo_sausage	-0.0549	0.2111	0.2401	-0.8017	0.1532
onion_flake	-0.1251	-1.3963	0.3539	0.3667	-0.4977
ground_beef_chuck	-0.2445	-0.4192	-0.4593	-1.5089	-0.7366
puff_pastry_shell	-0.0511	0.0009	-0.2038	-0.6524	-0.3916
saffron_thread	-0.2002	-0.7566	-1.0538	-0.6279	-1.5030
peach_preserves	0.2217	-0.1254	-0.3420	-0.6840	-0.6064
rock_lobster_tail	0.0183	-0.1151	0.0882	0.2742	-0.2557
Fontina_cheese	-0.1412	0.0273	0.1889	-0.1677	0.0845
bake	0.0954	-0.1974	-0.1715	-0.2504	-0.0997
simmer	-0.0047	0.9655	-0.0025	-2.0432	0.0727

Table 8: Full estimate on recipe data: IMML (K=4) and Mixed logit model

D. Figures

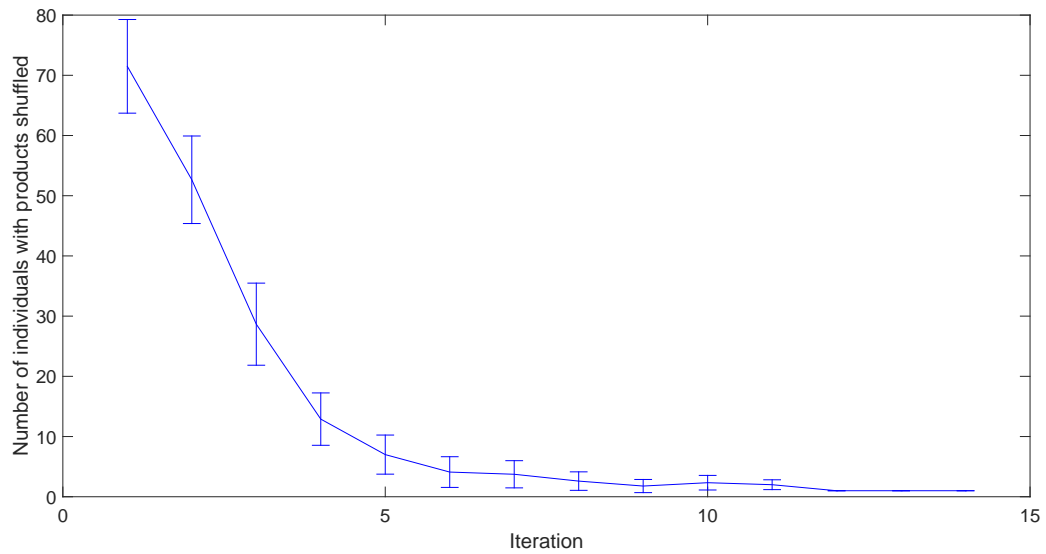


Figure 6 Mean (with standard deviation bar) number of individuals with products shuffled per iteration. It decreases with each iteration, indicating the algorithm converges.

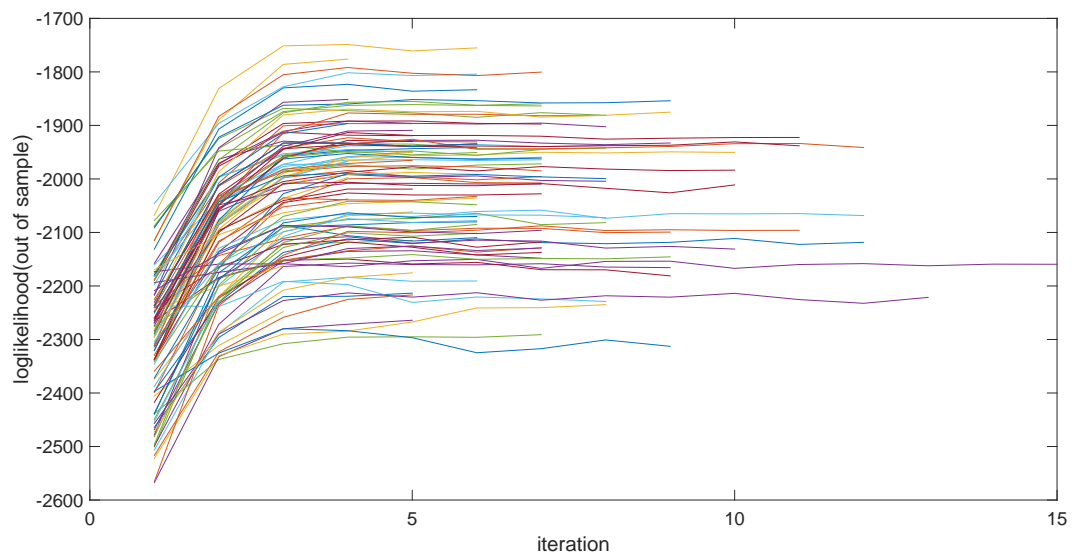


Figure 7 Likelihood (out-of-sample) per iteration. It decreases with each iteration, indicating the trained model is getting better each iteration in terms of out-of-sample likelihood.

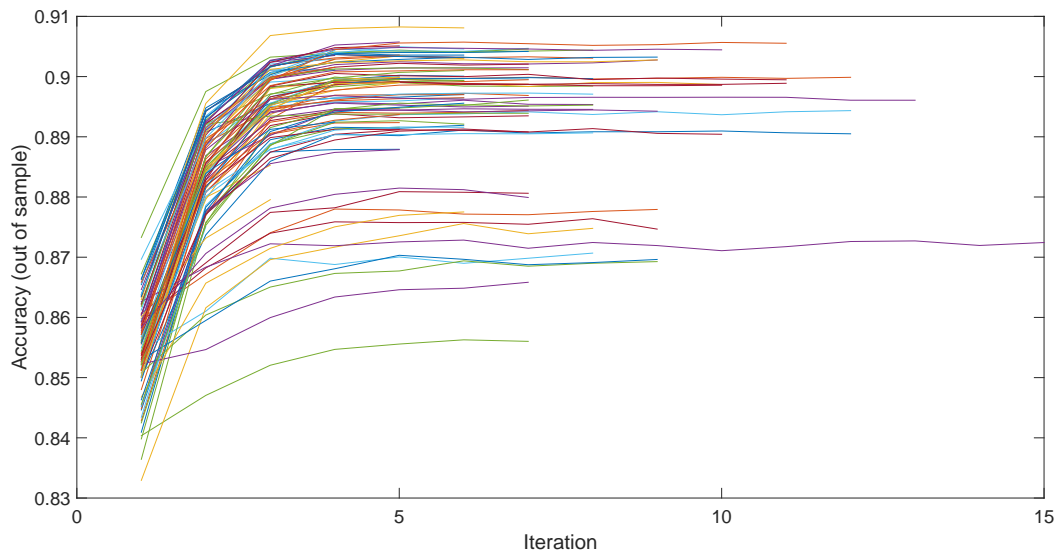


Figure 8 Accuracy (out-of-sample) per iteration. It decreases with each iteration, indicating the trained model is getting better each iteration in terms of out-of-sample accuracy.

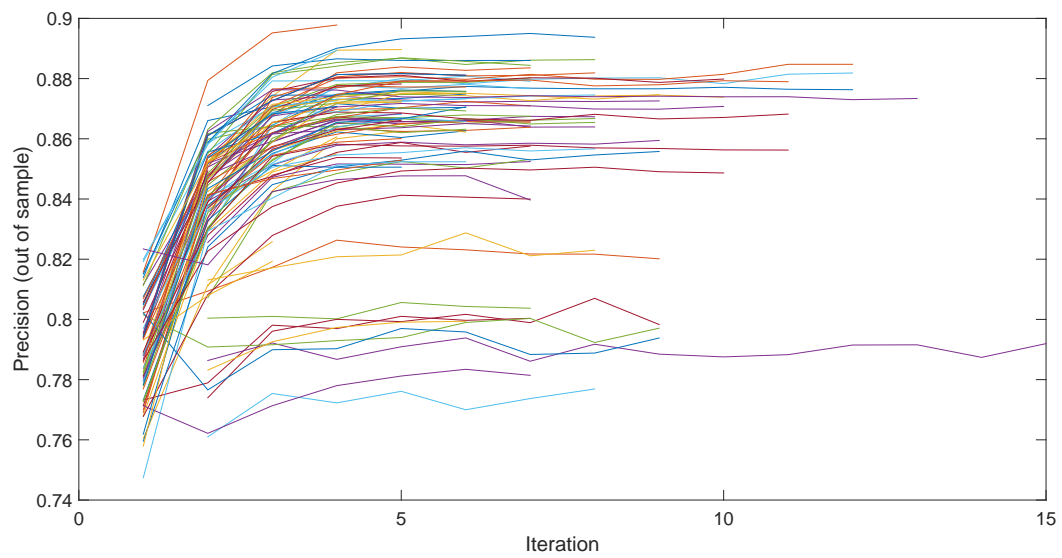


Figure 9 Precision (out-of-sample) per iteration. It decreases with each iteration, indicating the trained model is getting better each iteration in terms of out-of-sample precision.

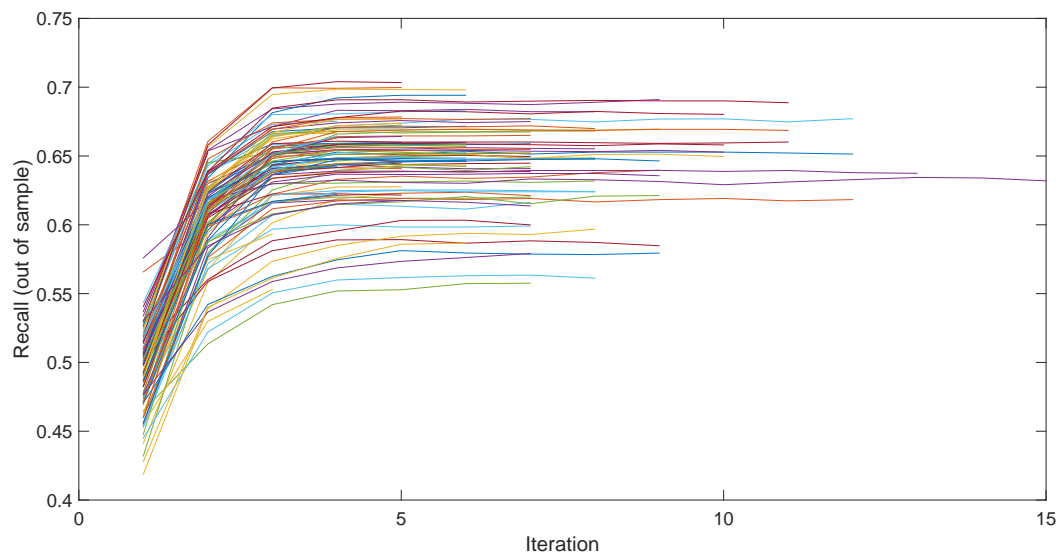


Figure 10 Recall (out-of-sample) per iteration. It decreases with each iteration, indicating the trained model is getting better each iteration in terms of out-of-sample recall.

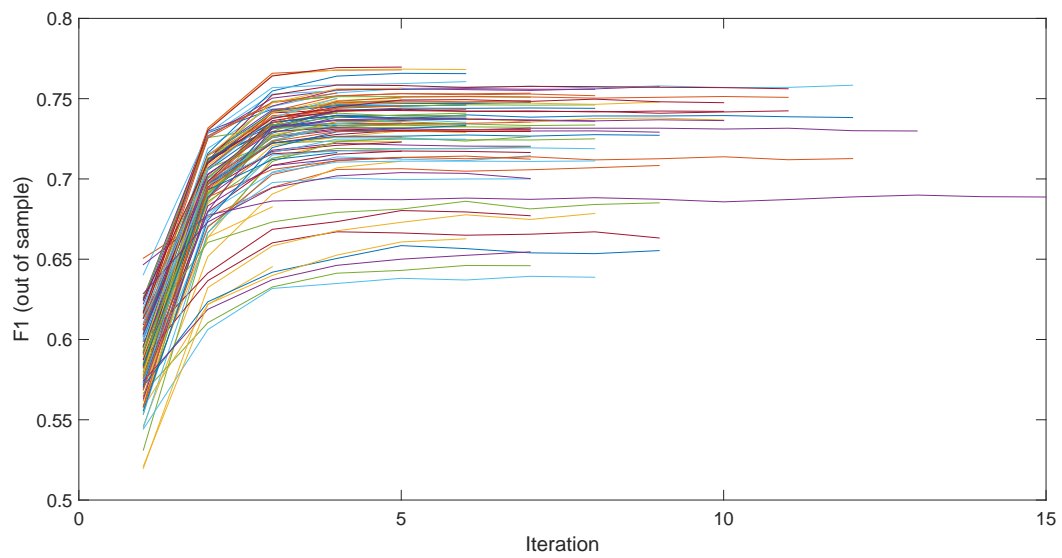


Figure 11 F1 (out-of-sample) per iteration. It decreases with each iteration, indicating the trained model is getting better each iteration in terms of out-of-sample F1.

References

- Allenby GM, Rossi PE (1998) Marketing models of consumer heterogeneity. *Journal of Econometrics* 89(1):57–78.
- Ansari A, Essegai S, Kohli R (2000) Internet recommendation systems. *Journal of Marketing Research* 37(3):363–375.
- Ansari A, Mela CF (2003) E-customization. *Journal of Marketing Research* 40(2):131–145.
- Bodapati AV (2008) Recommendation systems with purchase data. *Journal of Marketing Research* 45(1):77–93.
- Boxall PC, Adamowicz WL (2002) Understanding heterogeneous preferences in random utility models: a latent class approach. *Environmental and resource economics* 23(4):421–446.
- Bucklin RE, Gupta S (1992) Brand choice, purchase incidence, and segmentation: An integrated modeling approach. *Journal of Marketing Research* .
- Chintagunta PK (1992) Heterogeneity in nested logit models: an estimation approach and empirical results. *International Journal of Research in Marketing* 9(2):161–175.
- Chintagunta PK (1998) Inertia and variety seeking in a model of brand-purchase timing. *Marketing Science* 17(3):253–270.
- Chintagunta PK, Jain DC, Vilcassim NJ (1991) Investigating heterogeneity in brand preferences in logit models for panel data. *Journal of Marketing Research* 417–428.
- Chung J, Rao VR (2003) A general choice model for bundles with multiple-category products: Application to market segmentation and optimal pricing for bundles. *Journal of Marketing Research* 40(2):115–130.
- Danaher PJ, Mawhinney DF (2001) Optimizing television program schedules using choice modeling. *Journal of Marketing Research* 38(3):298–312.
- DeSarbo WS, Atalay AS, LeBaron D, Blanchard SJ (2008) Estimating multiple consumer segment ideal points from context-dependent survey data. *Journal of Consumer Research* 35(1):142–153.
- Dzyabura D, Hauser JR (2011) Active machine learning for consideration heuristics. *Marketing Science* 30(5):801–819.

- Evgeniou T, Boussios C, Zacharia G (2005) Generalized robust conjoint estimation. *Marketing Science* 24(3):415–429.
- Evgeniou T, Pontil M, Toubia O (2007) A convex optimization approach to modeling consumer heterogeneity in conjoint estimation. *Marketing Science* 26(6):805–818.
- Felzenszwalb P, McAllester D, Ramanan D (2008) A discriminatively trained, multiscale, deformable part model. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 1–8 (IEEE).
- Givon M (1984) Variety seeking through brand switching. *Marketing Science* 3(1):1–22.
- Greene WH, Hensher DA (2013) Revealing additional dimensions of preference heterogeneity in a latent class mixed multinomial logit model. *Applied Economics* 45(14):1897–1902.
- Guadagni PM, Little JD (1983) A logit model of brand choice calibrated on scanner data. *Marketing science* 2(3):203–238.
- Guadagni PM, Little JD (1998) When and what to buy: a nested logit model of coffee purchase. *Journal of Forecasting* 17(3-4):303–326.
- Hauser JR, Toubia O, Evgeniou T, Befurt R, Dzyabura D (2010) Disjunctions of conjunctions, cognitive simplicity, and consideration sets. *Journal of Marketing Research* 47(3):485–496.
- Huang D, Luo L (2012) Consumer preference elicitation of complex products using fuzzy support vector machine active learning. *revise and resubmit, Marketing Science. * AMA Advanced Research Techniques (ART) Forum Best Paper Award.*
- Kamakura WA, Russell G (1989) A probabilistic choice model for market segmentation and elasticity structure. *Journal of Marketing Research* 26:379–390.
- Kannan P, Wright GP (1991) Modeling and testing structured markets: A nested logit approach. *Marketing Science* 10(1):58–82.
- Kim JG, Menzefricke U, Feinberg FM (2007) Capturing flexible heterogeneous utility curves: A bayesian spline approach. *Management Science* 53(2):340–354.
- Koren Y, Bell R (2011) Advances in collaborative filtering. *Recommender systems handbook*, 145–186 (Springer).

- Lee JK, Sudhir K, Steckel JH (2002) A multiple ideal point model: Capturing multiple preference effects from within an ideal point framework. *Journal of Marketing Research* 39(1):73–86.
- Manchanda P, Ansari A, Gupta S (1999) The shopping basket: A model for multicategory purchase incidence decisions. *Marketing Science* 18(2):95–114.
- McFadden D, Train K, et al. (2000) Mixed mnl models for discrete response. *Journal of applied Econometrics* 15(5):447–470.
- Netzer O, Feldman R, Goldenberg J, Fresko M (2012) Mine your own business: Market-structure surveillance through text mining. *Marketing Science* 31(3):521–543.
- Rossi PE, McCulloch RE, Allenby GM (1996) The value of purchase history data in target marketing. *Marketing Science* 15(4):321–340.
- Rousseeuw PJ (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20:53–65.
- Seetharaman P, Chib S, Ainslie A, Boatwright P, Chan T, Gupta S, Mehta N, Rao V, Strijnev A (2005) Models of multi-category choice behavior. *Marketing Letters* 16(3-4):239–254.
- Singh VP, Hansen KT, Gupta S (2005) Modeling preferences for common attributes in multicategory brand choice. *Journal of Marketing Research* 42(2):195–209.
- Teng CY, Lin YR, Adamic LA (2012) Recipe recommendation using ingredient networks. *Proceedings of the 4th Annual ACM Web Science Conference*, 298–307 (ACM).
- Tibshirani R (1996) Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 267–288.
- Tirunillai S, Tellis GJ (2014) Mining marketing meaning from online chatter: Strategic brand analysis of big data using latent dirichlet allocation. *Journal of Marketing Research* 51(4):463–479.
- Train KE (2008) Em algorithms for nonparametric estimation of mixing distributions. *Journal of Choice Modelling* 1(1):40–69.
- Varki S, Chintagunta PK (2004) The augmented latent class model: Incorporating additional heterogeneity in the latent class model for panel data. *Journal of Marketing Research* 41(2):226–233.

- Weston J, Weiss RJ, Yee H (2013) Nonlinear latent factorization by embedding multiple user interests. *Proceedings of the 7th ACM Conference on Recommender Systems*, 65–68, RecSys '13 (New York, NY, USA: ACM).
- Ying Y, Feinberg F, Wedel M (2006) Leveraging missing ratings to improve online recommendation systems. *Journal of marketing research* 43(3):355–365.
- Yoganarasimhan H (2015) Search personalization using machine learning. *Available at SSRN 2590020* .
- Yu CNJ, Joachims T (2009) Learning structural svms with latent variables. *Proceedings of the 26th Annual International Conference on Machine Learning*, 1169–1176 (ACM).
- Yuille AL, Rangarajan A (2003) The concave-convex procedure. *Neural Computation* 15(4):915–936.
- Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67(2):301–320.