Partially-Ranked Choice Models for Data-Driven Assortment Optimization

Sanjay Dominik Jena
École des Sciences de la Gestion, Université du Québec à Montréal,
Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport (CIRRELT)
jena.sanjay-dominik@uqam.ca

Andrea Lodi
Polytechnique Montréal,
Canada Excellence Research Chair in Data-Science for Real-time Decision-Making,
Polytechnique Montréal
andrea.lodi@polymtl.ca

Hugo Palmer
BlaBlaCar, Paris
hugo.palmer@blablacar.com

Abstract

The assortment of products carried by a store has a crucial impact on its success. However, finding the right mix of products to attract a large portion of the customers is a challenging task. Several mathematical models have been proposed to optimize assortments. In particular, rank-based choice models have been acknowledged for representing well high-dimensional product substitution effects, and therefore reflect customer preferences in a reasonably realistic manner. In this work, we extend the concept of (strictly) fully-ranked choice models to models with partial ranking that additionally allow for indifference among subsets of products, i.e., on which the customer does not have a strict preference. We show that partially-ranked choice models are theoretically equivalent to fully-ranked choice models, but a partially-ranked preference sequence would require a factorial number of fully-ranked sequences to represent the same buying behavior. We then show how partially-ranked choice models can be learned efficiently from historical transaction and assortment data. The embedded column generation procedure involves subproblems that can be efficiently solved by using a growing decision tree that represents partially-ranked preferences, enabling us to learn preferences and optimize assortments for thousands of products. Computational experiments on artificially generated data and case studies on real industrial retail data suggest a significant potential to increase profits when performing data-driven assortment optimization and provide useful insights on customer segmentations to the decision makers, in our real case, the store managers. When comparing to existing algorithms, our method increases by one order of magnitude the scale of problems that can be learned by non-parametric choice models.

1 Introduction

Assortment planning denotes the process of identifying the set of products that should be offered to the customers. The planning problem is of paramount importance in operations and revenue management, since the choice of the assortment directly impacts the success of the business. However, from a managerial perspective, identifying the ideal assortment is a difficult challenge. While offering more products to the customer may eventually increase the number of sold items (also referred to as conversion), it is well known that an assortment that is too large may jeopardize the total sales. Of course, limitations of space and capacity may naturally limit the number of products the client will be exposed to. However, one may still observe that the presence of a certain product may decrease the sales of another, also known as cannibalization effect. In the same way, the absence of a product may encourage the customer to substitute to another (potentially more profitable) product, which illustrates the complexity of synergies among the products in an assortment.

The problem of finding the optimal assortment is crucial in several different domains. In particular, it is omnipresent in online advertisement on the internet, where it is necessary to decide for the limited number of advertisements that can be shown to a specific user profile in order to maximize the likelihood of conversion. In brick-and-mortar retail, assortment decisions are even more impactful, since the assortments are exposed to several different customer profiles and cannot be personalized to each customer type. Furthermore, changes in those assortments can be longsome and costly, given that products will have to be physically removed from the store and inventory and typically be liquidated at a far lower price. Mathematical models to help finding the "optimal" assortment are therefore finding increased popularity.

Defining a customer choice model that explains the market buying behavior sufficiently well is an essential step to optimize an assortment. Among the vast variety of choice models, rank-based choice models are rapidly gaining popularity. Rank-based choice models represent customer types via ranked preference lists on the available products. Those models hold a few important advantages. They can be easily interpreted by store managers and allow for insights regarding customer segmentation. Furthermore, they can be estimated in a purely data-driven manner without any assumptions on the market structure. In most of the application domains (such as online stores and even retail outlets), collecting transaction and assortment data has become a standard, therefore making such data sufficiently available. Data-driven models that automatically make assortment recommendations based on historical data and with limited user input are likely to dominate the operational planning of the retail industry in future. However, in practice, those models suffer from several challenges. On the computational side, rank-based choice models are hard to estimate. On the managerial side, even though preference lists (representing different customer types) provide store managers with certain insights, those lists typically contain unnecessarily large numbers (if not all) of products, which limits the usefulness of those lists to understand customer segmentation.

In this paper, we introduce a new representation for rank-based choice models. We argue that, next to strictly ranked products, customers may also be indifferent to a subset of the products and therefore buy any of those products with equal probability if their strictly preferred products are

unavailable. We show how we can learn this choice model efficiently by iteratively expanding a decision tree, in which each node implicitly represents a partially-ranked customer behavior. This approach is attractive from the computational point of view, since it converges quickly and allows to estimate choice models for large numbers of products within short computing times. At the same time, the method is appealing to store managers. The generated choice models are composed of a reasonably small number of different customer types. Moreover, the customer behaviors contain a significantly smaller list of ranked products that are necessary to explain the sales. Given that the final assortment is optimized via a *mixed-integer programming* (MIP) model, the modeler can easily integrate side-constraints such as capacities and requirements for different product categories (such as product subset or precedence constraints).

1.1 Relevant Literature

The essential foundation of practically effective assortment optimization is an appropriate choice model, which represents the buying behavior of the customers when faced with an assortment. Research on choice models dates back several decades. They have been applied to several domains such as transportation, marketing, and revenue management. The variety of different choice models is vast. While we will here focus on those that are most relevant in the context of our work, we refer the reader interested in broad overviews on assortment planning and choice models to surveys such as those of Mahajan and van Ryzin (1999) and Kök et al. (2008).

In the context of assortment optimization, two families of choice models (see, e.g. Jagabathula, 2011) have found predominant popularity in the literature and have been successfully applied in the aforementioned domains. The first family is that of (parametric) random utility maximization models. Among its most prominent members is the Multinomial Logit (MNL) choice model, which attributes an utility value to each of the available options. As in most of the choice models in revenue management, customers may also choose to abandon the buying process (e.g., if they are not willing to buy any of the available options), which is typically modelled as a dummy no-purchase options. The probability that a customer will then select a certain option increases with its utility relative to the sum of utilities of all options in the assortment. Even though these models are analytically and computationally tractable, they have several shortfalls. In particular, the MNL assumes the Independence of Irrelevant Alternatives (IIA) property (Arrow, 1951), due to which substitution effects among products (such as cannabilization) cannot be captured. Nested logit models, pioneered by Ben-Akiva (1973) for the modeling of travel demand, capture certain cases of substitution, but are still subject to the IIA property within each nest. Further extensions, such as Mixed Multinomial Logit (MMNL) models overcome those shortfalls and can capture quite general customer behaviors. Unfortunately, these models are computationally challenging for practical assortment optimization, given that they are typically non-linear and non-convex, in addition to the fact of involving discrete variables. Most importantly, parametric choice models generally rely on a good knowledge of the market structure and do strongly depend on the application context (see, e.g., Jagabathula, 2011), which makes them sensitive to issues of under- and over-fitting.

A second family of choice models has thus surged: non-parametric exponential models. Among them, rank-based models are quickly gaining popularity in the literature. Rank-based choice models generally assume that a customer behavior can be represented by a sorted preference list σ of the available options. The probability that a random customer follows the buying behavior according to a certain σ is given by a distribution over all possible preference sequences. The customer will then select the option that is ranked highest in her preference list and available in the assortment. Rank-based choice models have been acknowledged to offer manifold advantages. They implicitly capture high-dimensional substitution effects and therefore complex synergies among products. They are market uninformed and therefore not subject to over- or under-fitting, as it may be the case for parametric models. Further, their distribution can, theoretically, be derived from historical data. Such a purely data-driven approach is attractive from the manager perspective, as it requires little application-specific user input (often none at all) and is therefore easy to apply and to maintain. Having identified the relevant preference sequences may also give managers valuable insights about the customer segmentation.

Unfortunately, learning those choice-models over the space of different preference sequences, which is factorially large in the number of products N, is a major challenge. Both the identification of relevant customer behaviors and computing the underlying probability distribution that would explain the observed transactions are therefore computationally difficult. Some authors restrict the search space by relying on assumptions on the market structure. For example, Honhon et al. (2012) consider a special case that captures one level of substitution and therefore limits the search space to the order of N^2 customer behaviors. Recently, Vulcano and Van Ryzin (2017) provided efficient estimation methods when the number of different customer sequences is limited and known beforehand. However, in many settings, those assumptions may be overly restrictive, in particular when the customer market is not well known. In this regard, Jagabathula (2011) and Farias et al. (2013) strongly advanced research by circumventing the need for searching in factorially large space, without the requirement of a preinformed market structure. These authors consider only those models that minimize the revenue for a given assortment, therefore enabling the use of the dual problem and resulting in the choice model for worst-case prediction. An important feature of this approach is that it allows to identify the *sparsest* model, i.e., the choice model that is coherent with the observed transaction and requires the smallest number of different customer behaviors that have non-zero probability. From the managerial perspective, sparsity is strongly attractive, as it aims at explaining the customer market in the most condensed way possible. However, while the approach proposed by these authors may yield a good estimate of the worst-case revenue, it may not be ideal for managers, given that the assortment planning based on a worst-case choice model is likely to have a suboptimal performance on average. Bertsimas and Mišic (2016) proposed an algorithm to identify the preference sequences that best explain the sales out of the entire search space. The customer behaviors are identified via column generation and are then, together with their estimated probabilities, used in a mixed-integer programming model to provide an optimal assortment. Recently, the approach in Bertsimas and Mišic (2016) has been

further explored and improved by the same authors, see Bertsimas and Mišic (2017), to speed up the exact solution to the assortment optimization problem when the customer behaviors are given. Even though the assortment optimization scales fairly well, the computational complexity of identifying relevant customer behaviors is rather high. The entire process from choice model estimation (via transaction data) to the optimized assortment is therefore limited to rather small numbers of products. Ho-Nguyen and Kilinc-Karzan (2017) recently provided a uniform view of the methods discussed above. They provide a solution methodology to estimate rank-based choice models based on saddle point duality that particularly focuses on the context of dynamic learning, when choice models are updated with new data along time. Even though the authors provide a theoretical convergence guarantee for their algorithms, they do not present any numerical experiments.

We close this section by noting that, from a managerial perspective, one may also be interested in choice models that facilitate insights into the market segmentation and preferences. Again, sparsity is crucial, since it allows for separating the market into a smaller number of costumer types. However, this criterion is not considered in the approach of Bertsimas and Mišic (2016). Further, it is easy to see that products with low ranks have a marginally low probability of being selected. However, a customer behavior that consists of all possible products makes it difficult to understand which of those products are relevant in the sense that they actually contribute to explain sales. Therefore, another practically desired property is to generate preference lists that are as short as possible. We will denote this property as *concision*. In this work, we will introduce a new choice-model representation with the aim of efficiently estimating the choice models and provide assortment recommendations in a context where large numbers of products are available. At the same time, we aim at providing an approach that is attractive from a managerial perspective by providing choice models that are both sparse and concise.

1.2 Contributions

Our objective is to provide an operationally efficient methodology to estimate rank-based choice models without any assumptions on the market structure. Our contributions can be summarized as follows:

- *Partially-ranked choice model*: We introduce a new choice model representation, which conceptually generalizes the models of Farias et al. (2013) and Bertsimas and Mišic (2016). In addition to strictly ranked products, our model may additionally include *indifference sets*, which represent subsets of products that are selected by the customer with equal probability. It turns out that partially-ranked choice models can be seen as compact representations of fully-ranked choice models. We prove that both choice models are identical in the sense that, for any of those two choice models, one can identify an instance of the other choice model that represents an equivalent buying behavior. However, one partially-ranked customer behavior translates into a set of fully-ranked preference lists, which is of factorial size in the number of products in its indifference sets. The proposed choice model is therefore theoretically at least as sparse (i.e., it requires at most the same number of customer behaviors to explain

the sales) as the one used by Bertsimas and Mišic (2016). In computational experiments on synthetic instances, the partially-ranked choice models are on average about 4 times sparser.

- *Efficient estimation procedure*: We provide a novel efficient way to generate relevant customer behaviors by iteratively expanding a decision tree, in which each of the nodes represents a customer behavior. Finding new customer behaviors then amounts to expanding the tree, which can be done using column generation. While finding new columns with negative reduced costs can be generally costly, in our case, new columns can be found trivially thanks to the proposed tree structure.

- *Numerical results via simulation*: We provide numerical results on synthetic data that indicate the efficiency of our proposed methods. We further provide numerical results via simulations that explore the conditions for revenue increases, as well as the sparsity of the choice model and the concision of consumer behaviors.

- *Empirical study*: We report on a case study with real-world data from a major North-American fashion retail chain, provided by our industrial partner JDA Labs (JDA Labs, 2017).

1.3 Organization of the paper

The remainder of the paper is organized as follows. Section 2 introduces the new choice model and the procedure to efficiently identify relevant customer behaviors and the underlying distribution. Section 3 provides a mixed-integer programming model to provide an optimized assortment based on our choice model. Section 4 reports on numerical experiments from a simulation study on artifically generated data, and for real-world data from the retail industry. We conclude in Section 5.

2 A Partially-Ranked Choice Model

In this section, we will introduce a new representation for rank-based choice models that allows for strict preference on a subset of the products. This representation allows us to efficiently represent and construct relevant customer preferences using a decision tree. In the following, we will first introduce the new choice model and its representation as a tree. Then, a column generation will be used to grow the decision-tree and generate customer preferences. Finally, we will show how to use those developed customer preferences in the framework of an assortment optimization model.

Notation. We generally use bold faced characters such as $\boldsymbol{x} \in \mathbb{R}^N$ and $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ to represent vectors and matrices. We use x_i to denote the i-th element of vector $\boldsymbol{x}$.

2.1 The Choice Model

Rank-based choice models, such as the one introduced by Farias et al. (2013), assume that the market buying behavior is classified into different customer behaviors. Each customer behavior is represented by an ordered list that establishes a strict preference among all products. Preferred products are said to have *high ranks*, while less preferred products are said to have *low ranks*. The customer, following a specific behavior, buys exactly one product, which is the one that is *highest* ranked in her preference list and available in the presented assortment. Consider the set of products $\mathcal{N} = \{1, 2, \ldots, N\}$ and assume further that the customer always has the choice of selecting option 0, the no-purchase option, which would make her leave the store without any purchase. Following the notation used by Farias et al. (2013), we denote a customer behavior by σ and the rank of product i by $\sigma(i) \geq 0$. Consider a specific customer behavior σ with a list of preferred options $P(\sigma)$. A fully-ranked customer behavior can be defined as follows.

Definition 1 ***Fully-ranked customer behavior:*** *A customer behavior σ with a fully-ranked preference sequence may be written as a permutation of all $N + 1$ items in $\mathcal{N} \cup \{0\}$.*

As an example, consider 6 available products. The fully-ranked customer behavior $(3, 4, 1, 2, 5, 0, 6)$ indicates that the preferred product is 3, the second preferred product is 4, etc. A customer with such a preference will buy the highest ranked product that is available in the assortment. Note that product 6 will never be bought, since it is ranked after the no-purchase option 0.

A rank-based choice model $(\boldsymbol{\sigma}, \boldsymbol{\lambda})$ is then defined as a set $\mathcal{K} = \{\sigma_1, \ldots, \sigma_K\}$ of customer behaviors, together with a probability mass function $\boldsymbol{\lambda} \in \mathbb{R}^K$ that represents the corresponding probabilities that a random customer entering the store follows the specific behavior.

Rank-based choice models have several desirable properties, e.g., they allow for representing high-dimensional substitution effects. Recently, fully-ranked customer behaviors have also been used to perform subsequent optimization of the assortments (Bertsimas and Mišic, 2016). Despite their advantages, they are prone to some disadvantages. Generating fully-ranked customer preferences is computationally costly. Furthermore, from a management perspective, a fully-ranked sequence allows for little insights regarding customer segmentation. This is due to the fact that in reasonably large assortments, the probability that a product at, say, rank 10 will be selected, is rather low (see Proposition 1 further below). Most of the ranked products therefore explain only marginal portions of the sales (or none at all) and prevent store managers from identifying those products that actually have high impact (and explain sales).

More importantly, fully-ranked customer preferences may not be efficiently representing certain customer behaviors. In fact, it is hard to imagine that a customer has a strict preference order in mind that is defined on all products. Instead, she may rather have a strict preference order for a few of those products, but if none of those is available, she would choose any product from a subset of the available products with similar characteristics. As an example, consider a customer which has a specific sport shoes model in mind. If this model is not available in the exposed assortment, the customer may choose any other sport shoes model that has similar characteristics and is available

in the assortment. Clearly, only a subset of the products available in the assortment complies with these characteristics.

In the following, we will introduce a new representation for rank-based choice models that allows for a direct representation for subsets with equal sales probability. As we will see, this representation not only allows for a more efficient description of customer behaviors, but also allows for exploiting its structure and efficiently learning the description of the customer behaviors that are important to explain the sales.

Partially-ranked customer behaviors. The choice model proposed here assumes that customer preference lists do not necessarily have to impose a strict order on all products (which may be in the order of hundreds or thousands). Instead, a customer may have a strict preference on a subset of those products, e.g., 3, 4 and 1. If those products are absent in the assortment, the customer may buy any similar and available product, e.g., 2, 5 and 6. We may represent such a choice behavior as $\sigma = (P(\sigma), I(\sigma)) = (3, 4, 1, \{2, 5, 6\}, 0)$, where $P(\sigma) = (3, 4, 1) \subseteq \mathcal{N} \cup \{0\}$ is a strictly ranked list of preferred products and $I(\sigma) = \{2, 5, 6\} \subseteq \mathcal{N} \cup \{0\} \backslash P(\sigma)$ is the subset of *indifferent* products that will be chosen with uniform probability. There may be more than 6 products, but, assuming that product 0 is either in $P(\sigma)$, in $I(\sigma)$ or after $I(\sigma)$, those products will never be selected.

Definition 2 ***Simple partially-ranked customer behavior:*** *A simple customer behavior σ contains a strictly ranked preference lists $P(\sigma)$ and an indifference set $I(\sigma)$, where $P(\sigma) \subseteq \mathcal{N} \cup \{0\}$ and $I(\sigma) \subseteq \mathcal{N} \cup \{0\} \backslash P(\sigma)$ are mutually exclusive subsets of $\mathcal{N} \cup \{0\}$. If $0 \in P(\sigma) \cup I(\sigma)$, then the behavior can be written as $\sigma = (P(\sigma), I(\sigma))$; otherwise, $\sigma = (P(\sigma), I(\sigma), 0)$. Given an assortment S, the customer will select the product that is ranked highest in $P(\sigma)$ and available in S. If $P(\sigma) \cap S = \emptyset$, the customer will select any of the products in $I(\sigma) \cap S$ (which may include 0) with uniform probability. If $I(\sigma) \cap S = \emptyset$ and $0 \notin I(\sigma)$, the customer will leave the store.*

Again, note that it is not required to list all N products in $P(\sigma)$ or $I(\sigma)$, meaning that $P(\sigma) \cup I(\sigma) \subseteq \mathcal{N} \cup \{0\}$ and the inclusion can be strict. Each product from $\mathcal{N}$ can be part of only one between the strictly ranked list or indifference set, but the no-purchase option 0 has to be in either the list or the set, or at the end.

Even though the partially-ranked choice model seems to be more general than fully-ranked choice models, the underlying preference behaviors are essentially equivalent. Consider a simple partially-ranked customer behavior $(3, \{2, 5, 6\}, 0)$. The same choice model can be represented by a set of fully-ranked preferences, requiring one complete (fully-ranked) list for each of the permutations of the products in the indifference set: $(3, 2, 5, 6, 0)$, $(3, 2, 6, 5, 0)$, $(3, 5, 2, 6, 0)$, $(3, 5, 6, 2, 0)$, $(3, 6, 2, 5, 0)$ and $(3, 6, 5, 2, 0)$. This transformation holds for any simple partially-ranked preference list.

Lemma 1 *Consider a partially-ranked preference list as by Definition 2, containing one list of strictly ranked products and one indifference set, and occurring with probability λ_c. We can generate from σ_c a set with $|I(\sigma_c)|!$ fully-ranked preference lists and define the corresponding probabilities*

such that given the same assortment S, the final probability that product i is bought is the same in both cases.

Proof: Let S be a given assortment, $F = |I(\sigma_c)|$ and $G = |S \cap I(\sigma_c)|$. We generate $F!$ different fully-ranked preference lists, each containing the items in $P(\sigma_c)$ followed by a different permutation of the elements in $I(\sigma_c)$. For each of those fully-ranked lists, we define an equal probability $(\lambda_c/F!)$.

If $i \notin S$, the sales probability of i is 0 in both cases. If $i \in S$ and $i \in P(\sigma_c)$, then i will be bought with equal probabilities in both cases, since both the partially- and the fully-ranked lists start with the same sequence of products $P(\sigma_c)$.

For the case where $i \in S$ and $i \in I(\sigma)$ we need to show that the probabilities that i is bought given S are equal for the two cases (the partial preference list and the set of fully-ranked lists). For the partially-ranked list, the probability that product i is selected is defined as λ_c/G, i.e., the original probability divided by the number of products that are both in the indifference set and in the assortment. Recall that each of the generated fully-ranked lists has a probability of $(\lambda_c/F!)$. To compute the probability that i is selected, we multiply this probability by the number of fully-ranked lists in which i ranks highest in assortment S. The latter can be obtained in two steps. First, consider only the G items that are both in the assortment and in the indifference set and compute the number of permutations where i is ranked highest. It can be shown (e.g., via induction) that there are $(G-1)!$ permutations in which item i is at first rank (note that i has to be at the first rank to be selected, since all other $G-1$ items are also in the assortment). Then, for each of those $(G-1)!$ permutations, we compute the number of possibilities how to insert the remaining $(F-G)$ items (which are not in S) into the sub-list with G items. This can be computed by dividing the number of all permutations of the F items (those in $I(\sigma_c)$) by the number of permutations of the G already ordered items (which are in the assortment), i.e., $F!/G!$. The final probability that item i is selected given assortment S therefore amounts to $(\lambda_c/F!) \cdot (G-1)! \cdot (F!/G!) = \lambda_c/G$. ∎

More general than the simple partially-ranked customer behavior defined above, we may consider a customer behavior with q lists of preferred products $P^\ell(\sigma)$ $(\ell = 1, \ldots, q)$ and q indifferent sets $I^\ell(\sigma)$, defined as follows.

Definition 3 ***General partially-ranked customer behavior:*** *A general customer behavior σ has an ordered list of several strictly ranked preference lists $P^1(\sigma), \ldots, P^q(\sigma)$ and indifference sets $I^1(\sigma), \ldots, I^q(\sigma)$, all of which are mutually exclusive subsets of $\mathcal{N} \cup \{0\}$. If $0 \in \bigcup_{\ell=1,\ldots,q} P^\ell(\sigma) \cup I^\ell(\sigma)$, then the behavior can be written as $\sigma = (P^1(\sigma), I^1(\sigma), P^2(\sigma), I^2(\sigma), \ldots, P^q(\sigma), I^q(\sigma))$; otherwise, $\sigma = (P^1(\sigma), I^1(\sigma), P^2(\sigma), I^2(\sigma), \ldots, P^q(\sigma), I^q(\sigma), 0)$.*

Again, each product from $\mathcal{N}$ can be part of only one strictly ranked list or indifference set, and the no-purchase option has to be in one of the lists or sets, or at the end. Here, the customer would prefer to buy a preferred product within sequence $P^1(\sigma)$. If none of these products is available, the customer will choose any product with similar characteristics defined in $I^1(\sigma)$ and available in the assortment with uniform probability. If none of those products is available, further lists of preferred

products and sets of indifferent products may follow until the no-purchase option 0 indicates that no purchase is made. Recall the previous example of a simple partially-ranked customer behavior $\sigma = (P(\sigma), I(\sigma)) = (3, 4, 1, \{2, 5, 6\}, 0)$ with one strictly ranked product list and one indifference set. In the case that the customer does not find any of the indifferent products 2, 5 or 6 in the assortment, she may have further strict preferences (e.g., on a different product type), given by another strictly ranked product list, say $(7, 10, 9)$. If those products are also not available in the assortment, the customer may be indifferent on products 8 and 11. In the absence of those products, she may want to leave the store. This more complex behavior is represented by $\sigma = (3, 4, 1, \{2, 5, 6\}, 7, 10, 9, \{8, 11\}, 0)$, having two strictly ranked products lists and two indifference sets.

While fully-ranked customer preferences require a hierarchy among all products such that $\sigma(i) < \sigma(j)$ whenever product i is preferred to j, partially-ranked choice models also allow for relations of the form $\sigma(i) = \sigma(j)$, indicating that products i and j are equally preferred and therefore part of the same indifference set. The product ranks in the example above are as follows: $\sigma(3) = 0$, $\sigma(4) = 1$, $\sigma(1) = 2$, $\sigma(2) = \sigma(5) = \sigma(6) = 3$, $\sigma(7) = 4$, $\sigma(10) = 5$, $\sigma(9) = 6$, $\sigma(8) = \sigma(11) = 7$ and $\sigma(0) = 8$.

Based on Lemma 1, we now show that both the general partially-ranked choice model and the fully-ranked choice model can represent equivalent buying behaviors.

Theorem 2 ***Equivalence between fully-ranked and partially-ranked choice models:*** *Any choice model $(\boldsymbol{\sigma}^C, \boldsymbol{\lambda}^C)$ that contains fully-ranked customer behaviors (see Definition 1) can be represented by a choice model $(\boldsymbol{\sigma}^P, \boldsymbol{\lambda}^P)$ that contains only partially-ranked customer behaviors (see Definition 3). Further, any choice model $(\boldsymbol{\sigma}^P, \boldsymbol{\lambda}^P)$ that contains only partially-ranked customer behaviors can be transformed into an equivalent choice model $(\boldsymbol{\sigma}^C, \boldsymbol{\lambda}^C)$ exclusively composed of fully-ranked customer behaviors.*

Proof: To proof the first part, consider any preference list $\sigma_c \in \boldsymbol{\sigma}^C$. We may trivially derive an equivalent σ_p from σ_c by defining a corresponding partially-ranked customer behavior σ_p with the following parameters: $q = 1$, $P^1(\sigma_p) = P(\sigma_c)$, $I^1(\sigma) = \emptyset$ and $\lambda_p = \lambda_c$. We may do this for all behaviors in $\boldsymbol{\sigma}^C$ to derive the new choice model. To derive from $(\boldsymbol{\sigma}^P, \boldsymbol{\lambda}^P)$ an equivalent choice model composed of fully-ranked consumer behaviors, consider any $\sigma_p \in \boldsymbol{\sigma}^P$. It can be verified that the transformation of a simple partially-ranked customer behavior with one set of strictly ranked products and one indifference set in Lemma 1 can be generalized to a general partially-ranked customer behavior $(P^1(\sigma), I^1(\sigma), P^2(\sigma), I^2(\sigma), \ldots, P^q(\sigma), I^q(\sigma), 0)$ with several sets of strictly ranked products and several indifference sets. As in Lemma 1, we generate one fully-ranked list for each of the permutations of the products in the indifference sets, resulting in a total of $|I^1(\sigma)|! \cdot |I^2(\sigma)|! \cdot \ldots \cdot |I^q(\sigma)|!$ fully-ranked lists. Consider a product i and define r such that product i is either in $P^r(\sigma)$ or in $I^r(\sigma)$. The equivalence of the probability that i is selected is then proven in the same way as in Lemma 1 using $P^r(\sigma)$ and $I^r(\sigma)$. ∎

We may conclude from Theorem 2 that both, fully and partially-ranked choice models can

essentially reflect the same set of customer behaviors, but the latter can have a sparser representation. Choice models with full ranks can reflect preference indifference on products, but it will have to explicitly enumerate a number of fully-ranked lists that is factorial in the number of products in each of the indifferent sets. As outlined above, a general customer behavior $(P^1(\sigma), I^1(\sigma), P^2(\sigma), I^2(\sigma), \ldots, I^q(\sigma), 0)$ therefore requires $\prod_{q'=1}^{q} |I^{q'}(\sigma_P)|!$ fully-ranked sequences to represent an equivalent choice model. It is therefore desirable to find a more efficient representation for such customer behaviors that may significantly improve the tractability of training those models and performing subsequent optimization of future assortments. Partially-ranked behaviors are a promising candidate, in particular since low ranked products tend to have little impact in explaining sales, as stated below.

Proposition 1 ***Irrelevance of low ranked products:*** *The relevance of low ranked products can be negligibly small, both statistically and in practice. Consider a non-empty assortment $\mathcal{S}$, let $r = \frac{|\mathcal{S}|}{N} \in]0, 1]$ be the assortment density of $\mathcal{S}$ and assume that the probability that a certain product is part of $\mathcal{S}$ is uniform (i.e., it is equal to $\frac{|\mathcal{S}|}{N}$). Then, the relevance of a product decreases exponentially fast with its rank. Further, the greater the ratio r, the smaller the importance of low ranked products.*

Proof: The probability that the product at k^{th} rank is selected by the customer equals $(1-r)^{k-1} \cdot r$, where $(1-r)^{k-1}$ is the probability that none of the $k-1$ highest ranked products are selected and r is the probability that product k is subsequently selected. The result follows, as for any $k > 2$, the probability $(1-r)^{k-1} \cdot r$ reduces as r increases, and the decrease is exponential in k. ∎

With a ratio of $r = 0.1$, this probability is about 3.87% for rank $k = 10$, but only about 0.05% for $k = 50$. With higher ratios, lower ranks quickly become insignificant. For example, with $r = 0.5$, the probability for $k = 10$ is as low as 0.05%. While the above examples assume a uniform popularity in the customer priorities and in the assortments, in practice, it is likely that popular products are both highly ranked and are part of the assortment. We therefore argue that, in practice, the above stated probability tends to be a pessimistic (upper) bound on the importance of low ranked products. In other words, we expect that the probability that low ranked products are relevant decreases even more in practice. Further, an analysis of the industrial data used on 10 stores (see Section 4.2) showed that the total number of sold products is 192. It is reasonable to assume that stores carry assortments with at least 20 of those products, yielding a ratio r of at least 0.1. Our proposed methodology therefore explicitly focuses on the impact of the products at high ranks (i.e., those that are ranked as more preferred), reducing computational efforts, while preserving the accuracy of the choice models.

Before we introduce the method for estimating the proposed partially-ranked choice model, we will introduce a special case of the simple partially-ranked choice model (see Definition 2), where the indifference set contains all products that are not strictly ranked, i.e., $I(\sigma) = \mathcal{N} \cup \{0\} \backslash P(\sigma)$.

Definition 4 ***Partially-ranked customer behavior with complementary indifference set:*** *A partially-ranked customer behavior σ with complementary indifference set is defined as a simple partially-ranked customer behavior (see Definition 2), and imposing that $I(\sigma) = \mathcal{N} \cup \{0\} \backslash P(\sigma)$.*

While all further developments also carry over to the choice models composed by the most general partially-ranked customer behaviors as specified by Definition 3, we will restrict our attention from now on mostly to the simplified case given by Definition 4. The use of this simplified case will facilitate the comprehension of the proposed algorithms and is further justified by several other reasons. Even though using an indifference set that is composed of all non-ranked products may seem unrealistic in practice, one needs to keep in mind that such a consumer behavior can be transformed into an equivalent set of fully-ranked consumer behaviors (see Theorem 2). Further, this type of indifference set holds two important advantages. First, it allows for a very efficient computation of the reduced costs within the column generation scheme, which will be shown in the next section. Second, given that each partially-ranked list with an indifference set is equivalent to a large (in fact, factorially large in the size of the indifference set) number of fully-ranked lists, the products in the indifference set provide a quite direct, both granular and aggregated, explanation of sales when estimating the choice model (see Section 2.2). The products in the indifference set therefore hold a certain explanatory power, which can be computed exactly in an average assortment, as stated in the following proposition.

Proposition 2 ***Explanatory power of the indifference set:*** *Consider a consumer behavior $\sigma_k = (P(\sigma_k), I(\sigma_k))$ with estimated market probability λ_k and an assortment $\mathcal{S}$ with products density r. The contribution of σ_k to the explanation of the overall sales of product $i \notin P(\sigma_k)$ in $\mathcal{S}$ amounts to $(1-r)^{|P(\sigma_k)|} \frac{\lambda_k}{|I(\sigma_k)|}$.*

Proof: There are $|\mathcal{S}| - |P(\sigma_k)|$ products in the indifference set $I(\sigma_k)$. If none of the strictly ranked products is part of the assortment, the contribution to explaining the sales of one of the products in $I(\sigma_k)$ is $\frac{\lambda_k}{|I(\sigma_k)|}$. Further, the probability that none of the strictly ranked products is part of the assortment is, on average, $(1-r)^{|P(\sigma_k)|}$ (compare Proposition 1). The total average contribution of σ_k to a product $i \in I(\sigma_k)$ follows as the product of the previous two terms. ∎

As a consequence, one can easily verify the average explanatory impact on products that are not strictly ranked in a partially-ranked consumer behavior. For example, if $N = 100$, $|\mathcal{S}| = 50$ (thus, $r = 0.5$), 5 products are strictly ranked (i.e., $|P(\sigma_k)| = 5$, $|I(\sigma_k)| = N - |P(\sigma_k)| = 95$) and $\lambda_k = 0.05$, σ_k explains only 0.0016% $(= (1-0.5)^5 \cdot \frac{0.05}{95})$ of the sales of any product $i \in I(\sigma_k)$. The impact of using such an indifference set on explaining the overall sales, for this example, amounts to 3.12% $(= (1-0.5)^5)$. Therefore, exposing the decision maker to such a concise list of 5 products is practically sufficient, i.e., exhibiting the 5 products that explain 96.78% of sales caused by this consumer type.

2.2 Learning Consumer Preferences

Before we can optimize future assortments, we need to estimate the probability $\mathbb{P}(i|\mathcal{S})$ that a product i is sold given that a random customer is exposed to assortment $\mathcal{S}$. We may compute this probability, once our choice model $(\boldsymbol{\sigma}, \boldsymbol{\lambda})$ is estimated. Given that there is factorial number of different customer behaviors, a major challenge is to identify the set $\boldsymbol{\sigma}$ that is relevant for explaining the sales, as well as their corresponding probabilities $\boldsymbol{\lambda}$. In this section, we focus on how to efficiently learn those parameters that are consistent with the observed sales.

Assume that historical data is available including a total of M assortments, given by set $\mathcal{M} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M\}$, as well as sales transaction data for each of them. Those sales are given in a vector $\boldsymbol{v} \in \mathbb{R}^{(N+1)\cdot M}$ consisting of all pairs (i, m), with $i \in \mathcal{N} \cup \{0\}$, $m \in \mathcal{M}$, which represent the customers that have chosen option i when being presented assortment S_m. Note that it is assumed that such sales data also includes the no-purchase option 0, e.g., if it is accurately collected by the store or estimated by the store manager. We are then concerned with learning a choice model $(\boldsymbol{\sigma}, \boldsymbol{\lambda})$ that is consistent with sales probabilities $\boldsymbol{v}$. Assume that one has identified a set of relevant customer behaviors $\mathcal{K}$. One may then compute a matrix $\boldsymbol{A} \in \mathbb{R}^{((N+1)\cdot M)\times K}$, in which an entry $A^k_{(i,m)}$ is set to 1 if customer k would choose product i from assortment $\mathcal{S}_m \cup \{0\}$. As a consequence, $\forall (k, m) : \sum_i A^k_{(i,m)} = 1$.

Similar to Bertsimas and Mišic (2016), we will minimize the L_1 error between $\boldsymbol{v}$ and $\boldsymbol{A\lambda} = \sum_k A^k_{(i,m)} \lambda_k$, the probability that a random customer chooses option i from assortment m. To avoid dependence from the number of assortments, we further propose to normalize the error, and therefore minimize:

$$\epsilon^{Tr} = \frac{1}{2|M|} \sum_{(i,m)} |\boldsymbol{A\lambda} - \boldsymbol{v}|_{i,m}.$$

Consider a given set of potentially relevant customer behaviors $\{\sigma^1, \sigma^2, ..., \sigma^K\}$, one may use a simple linear program to find the corresponding probability distribution $(\lambda_1, \lambda_2, ..., \lambda_K)$ that results in the smallest error as follows:

$$\min_{\boldsymbol{\lambda}, \boldsymbol{\epsilon}^+, \boldsymbol{\epsilon}^-} \quad \mathbf{1}^T \boldsymbol{\epsilon}^+ + \mathbf{1}^T \boldsymbol{\epsilon}^- \tag{1a}$$

$$\text{s.t.} \quad \boldsymbol{A\lambda} + \boldsymbol{\epsilon}^+ - \boldsymbol{\epsilon}^- = \boldsymbol{v} \tag{1b}$$

$$\mathbf{1}^T \boldsymbol{\lambda} = 1 \tag{1c}$$

$$\boldsymbol{\lambda}, \boldsymbol{\epsilon}^+, \boldsymbol{\epsilon}^- \geq 0. \tag{1d}$$

Setting the Choice Matrix A. Bertsimas and Mišic (2016) propose to set the choice matrix $\boldsymbol{A}$ for fully-ranked customer behaviors such that $\boldsymbol{A\lambda} = \boldsymbol{v}$ by using entry 1 if customer k would choose product i from $\mathcal{S}_m \cup 0$. This notion does not hold in the context of our more general representation which may include indifference sets. For reasons of simplicity and without loss of generality, let us consider the slightly simpler case $(P(\sigma), I(\sigma), 0)$ in which the customer behavior σ consists only of a single list $P(\sigma)$ of preferred and strictly ranked products, followed by an indifference set $I(\sigma)$.

We assume that the indifference set $I(\sigma)$ (in the more general case, we have $I^1(\sigma), \ldots, I^q(\sigma)$) is externally given, for instance by a marketing department. Alternatively, it can be learned or estimated in a previous step by identifying products with similar characteristics. We may then define the ranking entries of a customer behavior σ as follows:

$$\sigma(i) = \begin{cases} \text{rank of preference of } i & \text{if } i \in P(\sigma) \\ |P(\sigma)| & \text{if } i \in I(\sigma) \\ +\infty & \text{otherwise.} \end{cases}$$

The preferred products are ranked from 0 to $|P(\sigma)| - 1$, whereas all products in the indifference set have equivalent rank $|P(\sigma)|$. In the general case with several indifference sets, those ranks will be computed as the rank of the previous preferred product plus 1. The rank of a product that is neither in a $P(\sigma)$ nor in $I(\sigma)$ is set to $+\infty$.

Setting the choice matrix $\boldsymbol{A}$ for a partially-ranked choice model has to respect $\forall(k,m)$: $\sum_i A^k_{i,m} = 1$, which we can achieve by a uniform distribution on the indifference set:

$$A^k_{i,m} = \begin{cases} 1 & \text{if } i \in \mathcal{S}_m \text{ and } \forall j \in \mathcal{S}_m \backslash \{i\} : \sigma^k(i) < \sigma^k(j), \\ \dfrac{1}{|I(\sigma) \cap \mathcal{S}_m|} & \text{if } i \in \mathcal{S}_m : \text{ and } \forall j \in \mathcal{S}_m, \sigma^k(j) = |P(\sigma)| \text{ or } \sigma^k(j) = +\infty, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The choice matrix $\boldsymbol{A}$ can therefore be efficiently computed taking into consideration any general customer behavior σ that is consistent with Definition 3.

Learning Consumer Behaviors based on a Decision Tree. As noted by Bertsimas and Mišic (2016), the linear program (1) is not tractable, given the factorial number of customer behaviors here considered, causing $\boldsymbol{A}$ and $\boldsymbol{\lambda}$ to be exponentially large. The authors therefore propose a column generation algorithm, which initializes problem (1) as Master problem with a small subset of promising columns (i.e., preference sequences). The algorithm then iteratively identifies possibly relevant columns and adds them to the Master problem. Let $\boldsymbol{\alpha}$ and ν be the dual values of constraints (1b) and (1c) after solving problem (1). To find columns with minimal reduced cost, the authors further propose the following mixed-integer program:

$$\min_{z,a} \ \boldsymbol{\alpha}^T \boldsymbol{a} - \nu \tag{3a}$$

$$\text{s.t.} \ \ a_{i,m} \leq z_{ij} \qquad \forall m \in \{1, ..., M\},\ i, j \in S_m \cup \{0\},\ i \neq j \tag{3b}$$

$$z_{ij} + z_{ji} = 1 \qquad \forall i, j \in \{0, 1, ..., N\},\ i \neq j \tag{3c}$$

$$z_{ij} + z_{j\ell} - 1 \leq z_{i\ell} \qquad \forall i, j, \ell \in \{0, 1, ..., N\}\ i \neq j,\ i \neq \ell,\ j \neq \ell \tag{3d}$$

$$\boldsymbol{z} \in \{0, 1\},\ \boldsymbol{a} \in \{0, 1\}.$$

The MIP (3) minimizes the total reduced costs of the corresponding product sequence defined by the $\boldsymbol{z}$ variables. Constraints (3b) ensure that $a_{i,m}$ can take a positive value only if product i is preferred to all other products in assortment m. The set of constraints (3c) and (3d) represent non-reflexivity and transitivity, respectively, in order to establish a strict order among all products. Unfortunately, system (3) is costly to solve, and one needs to find alternatives for practical purposes. The authors therefore use a local-search heuristic to find new columns with reduced costs. Both the heuristic and the exact model (3) generate fully-ranked customer behaviors, which may not be necessary in practice (see Proposition 1). We therefore propose to focus on products with high ranks (i.e., those that are considered more preferred) and to explicitly take advantage of the structure of the proposed choice model.

The partially-ranked customer behaviors with indifference subsets can be efficiently represented by a decision tree, in which explicitly listed nodes refer to strictly ranked products. In the general case (see Definition 3) with several indifferent sets that are strict subsets of N, each indifferent set can be represented by a node that includes several products. However, selecting those sets may be context specific and require input from store managers on how to cluster those products. While the presented methods also apply to the general case of partially-ranked customer behaviors (see Definition 3), we will focus, without loss of generality, on the simpler case of $\sigma = (P(\sigma), I(\sigma))$ as specified in Definition 4. Here, the indifference set contains all nodes that are not strictly ranked, i.e., $I(\sigma) = N \backslash P(\sigma)$, and therefore does not need to be explicitly listed in the tree. Figure 1 illustrates a small example with 3 products and the no-purchase option 0. In this example, a total of $|K| = 8$ customer behaviors has been generated. For instance, customer behavior σ_7, associated with node λ_7, refers to a customer that prioritizes product 2, if present; if not, she is willing to buy product 1. If none of those products is available, the customer will buy any available product or leave the store with equal (i.e., uniform) probability. In contrast, customer behavior σ_6, associated to node λ_6, refers to a customer that will buy product 2, if available, and leave the store without purchase otherwise (indicated by the no-purchase option 0).

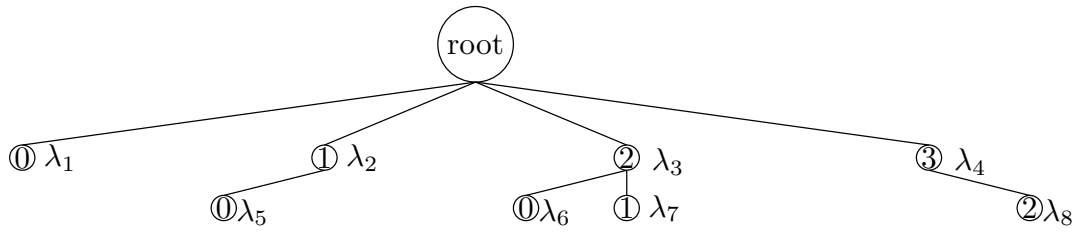


Figure 1: Example of Growing Decision Tree choice model for $N = 3$ products

It becomes immediate that such a tree structure is a quite intuitive representation for store managers who may want to understand customer segmentation. It tends to generate a smaller number of customer behaviors and focuses on the products that are important to explain sales: those that are ranked early in a customer preference sequence. Further, the search for new customer behaviors in the tree structure may drastically speed up computation if one succeeds to limit the

search to a significantly smaller space. This is the case if we focus on high ranks first and then gradually expand the tree by not more than one level of depth at each branch and iteration. Due to the gradual expansion of the tree, we will refer to it as the *Growing Decision Tree (GDT)*.

We define σ_j to be a *sub-behavior* of σ_i if $P(\sigma_j) = (P(\sigma_i), \ell)$, where $\ell \in I(\sigma_i)$. In words, a sub-behavior σ_j inherits the strict preference list from its parent σ_i and adds to it one product (including, potentially, the no-purchase option) that is not part of σ_i's preference list. Let $\mathcal{K}$ be the set of behaviors enumerated in the GDT. Our algorithm iteratively searches for new sub-behaviors in the GDT, expands the tree and solves problem (1) to find the corresponding probabilities $\boldsymbol{\lambda}$. When looking for new promising columns (i.e., new sub-behaviors), we may restrict the search to the sub-behaviors of all $\sigma \in \mathcal{K}$. The reduced costs of each of the sub-behaviors can be computed as $rc(\sigma) = -\boldsymbol{\alpha}\boldsymbol{a} - \nu$, where $\boldsymbol{\alpha}$ and ν are the dual values of constraints (1b) and (1c) in problem (1), and a is defined according to equality (2). The sub-behaviors with the lowest reduced cost are then added to the set of customer behaviors $\mathcal{K}$, and problem (1) is resolved. The pricing step is exemplified in Figure 2, in which the reduced cost for all sub-behaviors (indicated in blue) of behaviors $\sigma \in K$ are computed, unless the last product in the preference list of σ is option 0. This would indicate that the customer would leave the store and the sub-behaviors are irrelevant. The process is performed iteratively until the L_1 error is sufficiently small or a defined maximum number of iterations is performed.

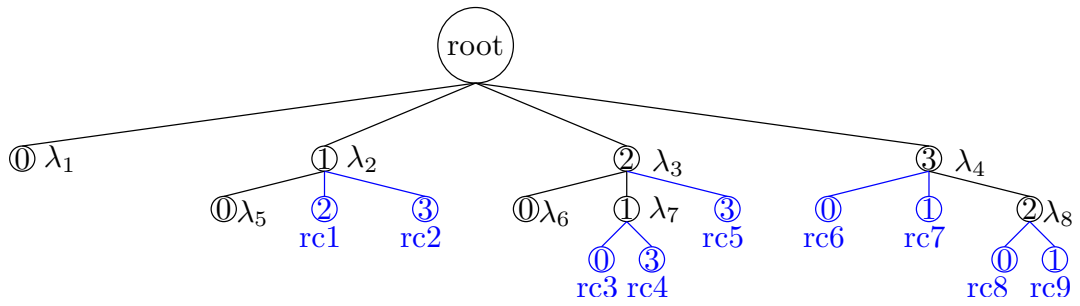


Figure 2: Computing reduced costs in the Growing Decision Tree choice model

Note that computing the reduced cost of a fully-ranked preference list has a computational complexity of $O(N)$, even if the reduced cost of a "similar" fully-ranked preference list is known. Therefore, finding new columns with negative reduced costs can become costly when using fully-ranked preference lists. In contrast, using partially-ranked preference lists and the GDT holds the remarkable advantage that, once the reduced cost of a partially-ranked list is known, the reduced cost of any of its sub-behaviors can be computed in constant time. The reduced cost of the sub-behavior is similar to the one of the current column, given that the sub-behavior ranks exactly one more product (adding the corresponding dual value α to the reduced cost) and therefore has one item less in the indifference set (subtracting from the reduced cost the corresponding dual value α divided by the number of elements in the previous indifference set). This makes the exploration of the search space for negative reduced cost columns extremely efficient.

Algorithm 1 outlines the complete column generation procedure to build the GDT. At each

Algorithm 1: GDT-based column generation algorithm

Input data :

- Sales probability vector $\boldsymbol{v}$, training set of assortments $S_1,\ldots,S_M$.
- Maximum number of column generation iterations $iter^{CG}_{MAX}$.
- Optimality training criteria ϵ^{Tr}_{MIN}.
- Maximum number of sub-behaviors δ that are added at each iteration.
- Maximum depth $\overline{d}$ to find sub-behaviors with negative reduced cost

Output data:

- A set $\mathcal{K} = \{\sigma_0, \ldots, \sigma_{K-1}\}$ of K customer behaviors, where $\sigma_k = (P(\sigma_k), I(\sigma_k), 0)$.

begin
  Initialize set $\mathcal{K}$ with $N+1$ sub-behaviors $\sigma_k,\ k = 0,\ldots,N$, defined with $P(\sigma_k) = (k)$ and $I(\sigma_k) = \mathcal{N} \setminus \{k\}$.
  Set *iter* to 0.
  Solve restricted Master Problem (1) to obtain $\boldsymbol{\lambda}$, $\boldsymbol{\epsilon}^+$, $\boldsymbol{\epsilon}^-$ and dual values $\boldsymbol{\alpha}$ and ν.
  while $(iter \leq iter^{CG}_{MAX})$ *and* $(\mathbf{1}^T\boldsymbol{\epsilon}^+ + \mathbf{1}^T\boldsymbol{\epsilon}^- > \epsilon^{Tr}_{MIN})$ **do**
    Set $iter \longleftarrow iter + 1$
    Set $depth \longleftarrow 0$
    Set $\mathcal{C}^P \longleftarrow \mathcal{K}$
    Set $\mathcal{C}^N \longleftarrow \emptyset$
    Set $\mathcal{D} \longleftarrow \emptyset$
    while $(\mathcal{D} = \emptyset)$ *and* $(depth \leq \overline{d})$ **do**
      Set $depth \longleftarrow depth + 1$
      for $\forall \sigma_k \in \mathcal{C}^P$ **do**
        if $P(\sigma_k)_{|P(\sigma_k)|} \neq 0$ *(i.e., if last element in $P(\sigma_k)$ is not 0)* **then**
          compute the reduced costs for all new sub-behaviors of σ_k
          Add each new sub-behavior with negative reduced cost to $\mathcal{C}^N$
      if $(\mathcal{C}^N = \emptyset)$ **then**
        Set $\mathcal{C}' \longleftarrow \emptyset$
        for $\forall \sigma_k \in \mathcal{C}^P$ **do**
          Add all sub-behaviors of σ_k to $\mathcal{C}'$
        Set $\mathcal{C}^P \longleftarrow \mathcal{C}'$
      else
        Add to $\mathcal{D}$ up to δ sub-behaviors $\sigma_k \in \mathcal{C}^N$ that have the lowest reduced costs
    if $(\mathcal{D} = \emptyset)$ **then**
      Solve MIP (3) to find σ_k with smallest reduced cost
      if *(σ_k's reduced cost is negative)* **then**
        Add σ_k to $\mathcal{D}$
      else
        Return $\mathcal{K}$
    Set $\mathcal{K} = \mathcal{K} \cup \mathcal{D}$ and all sub-behaviors $\in \mathcal{D}$ as new columns to matrix $\boldsymbol{A}$
    Solve restricted Master Problem (1) to obtain $\boldsymbol{\lambda}$, $\boldsymbol{\epsilon}^+$, $\boldsymbol{\epsilon}^-$ and dual values $\boldsymbol{\alpha}$ and ν
  return $\mathcal{K}$;

iteration, the algorithm first explores the sub-behaviors of the behaviors in $\mathcal{C}^P$, which represents all behaviors that have positive reduced costs at the current iteration. If sub-behaviors with negative reduced costs are found, they are added to the set of columns with negative reduced costs $\mathcal{C}^N$. If none of the explored sub-behaviors has reduced cost, the algorithm replaces all behaviors in $\mathcal{C}^P$ by their respective sub-behaviors and therefore continues the search for columns with negative reduced costs on a deeper level of the tree. The algorithm continues until either a maximum depth $\overline{d}$ has been reached or at least one column of negative reduced cost has been found. If, after having explored depth $\overline{d}$ no new column with negative reduced cost has been found, MIP (3) can be employed to find the most negative reduced cost column. If this cost is positive, optimality has been proven and the algorithm terminates. Otherwise, the column is added and the procedure is continued.

The column generation procedure, as outlined in Algorithm 1, explores the sub-behaviors of all $\sigma_k \in \mathcal{C}^P$ (see code lines 13 and 19). In practice, exploring all nodes may be unnecessarily time consuming. Instead, we may randomly select up to γ behaviors for which the sub-behaviors should be explored. Lines 13 to 21 then have to be iteratively performed until a maximum number of those iterations is explored and the algorithm proceeds to the next depth. In our computational experiments, we set the maximum depth $\overline{d} = 1$. We also set $\gamma = 5$ and select those behaviors randomly, weighted by their probabilities λ_k.

Contrary to using MIP (4) to identify the customer behavior with the lowest reduced cost, the GDT allows for controlling which type of customer behaviors to consider. For example, if indifferent sets are not at all desired by the modeler, one only needs to consider the sub-behaviors that end in the no-purchase option 0 and set matrix A accordingly. The GDT would then only generate strictly ranked customer behaviors, but most likely converge much faster than when using the MIP (4) or a local search.

3 Assortment optimization

In the previous sections, we have presented a new representation for rank-based choice models and an efficient methodology to identify a set $\mathcal{K}$ of relevant customer behaviors $\sigma_k \in \mathcal{K}$, as well as their corresponding probabilities λ^k. We now focus on how to identify optimal assortments that are coherent with the learned choice model. Aouad et al. (2015) recently showed that assortment optimization based on a given choice model is generally NP-hard. For ranked-based models, an early MIP formulation has been introduced by Belloni et al. (2008), which was limited to small problem instances. Nevertheless, some recent works have proposed optimization models that scale reasonably well (see Bertsimas and Mišic, 2016, 2017; Farias et al., 2013).

We suppose that a revenue r_i is associated with each product i. The no-purchase option 0 yields a revenue of 0. Bertsimas and Mišic (2016, 2017) propose a mixed-integer programming model that uses variables x_i that take value 1 if product i is included in the assortment, and 0 otherwise. It further uses variables y_i^k that take value 1 if product i is within the assortment and is chosen

according to behavior σ_k. The optimization problem writes as

$$\max_{x,y} \quad \sum_{k=1}^{K} \sum_{i=1}^{N} r_i \lambda^k y_i^k \tag{4a}$$

$$\text{s.t.} \quad \sum_{i=0}^{N} y_i^k = 1 \qquad \forall k \in \{1, ..., K\} \tag{4b}$$

$$y_i^k \leq x_i \qquad \forall k \in \{1, ..., K\},\ \forall i \in \{1, ..., N\} \tag{4c}$$

$$\sum_{j:\sigma^k(j)>\sigma^k(i)} y_j^k \leq 1 - x_i \quad \forall k \in \{1, ..., K\},\ \forall i \in \{1, ..., N\} \tag{4d}$$

$$\sum_{j:\sigma^k(j)>\sigma^k(0)} y_j^k = 0 \qquad \forall k \in \{1, ..., K\} \tag{4e}$$

$$\boldsymbol{x} \in \{0, 1\},\ \boldsymbol{y} \geq 0.$$

The optimization problem (4) maximizes the expected revenue. Constraints (4b) select exactly one product for each customer type k. Constraints (4c) say that a product can be selected only if it is part of the assortment. Constraints (4d) guarantee that the product in the assortment that is ranked highest also has the highest y_i^k value. Finally, constraints (4e) ensure that all products ranked lower than the no-purchase option 0 are not selected.

Problem (4) is highly tractable. In particular, even though defined as continuous variables, variables $\boldsymbol{y}$ will only take binary values due to the structure of the problem. Bertsimas and Mišic (2017) explicitly showed that the formulation yields stronger linear programming relaxation bounds than the formulation initially proposed by Belloni et al. (2008). Unfortunately, the formulation requires fully-ranked customer behaviors, which are not explicitly given by a choice model with partially-ranked behaviors as generated by the GDT column generation algorithm. To adapt partially-ranked behaviors to the MIP stated above and obtain an exact approach, we could transform the partially-ranked choice model into a fully-ranked choice model. However, the number of necessary fully-ranked preference lists is factorially large (see Theorem 2), which would make the resulting MIP intractably large. As an heuristic approximation, one may replace the indifference sets by a random sequence of those products that are not strictly ranked. Generating several random sub-behaviors for each partially-ranked customer's behavior, generally known as *boosting*, may be computationally tractable while improving the performance of the final assortments. However, it may be quite instance specific how many of those random sequences to generate, and the models may become too large anyhow, without mentioning the heuristic nature of the method.

We are therefore interested in finding an optimization model that directly operates on partially ordered ranks, i.e., those that use a strict ranking on a subset of products and indifference on the remaining products. To directly operate on customer behaviors with indifference sets, we may add the following constraints, which enforce that y variables for products i and j have equal values if

both products are part of the assortment and have equivalent rank in behavior k. Namely,

$$z_{ij} = x_i \cdot x_j \quad \forall i, j \in \{1, \ldots, N\} : i > j \tag{5}$$

$$|y_i^k - y_j^k| \leq 1 - z_{ij} \quad \forall k \in \{1, ..., K\};\ \forall i, j \in \{1, \ldots, N\} : i > j \text{ and } \sigma^k(i) = \sigma^k(j). \tag{6}$$

The introduction of variables z_{ij} and the linearization of constraints (5) and (6) would significantly increase the model size and the difficulty of solving the problem. Fortunately, an equivalent model can be achieved by adequate transformation and substitution of the new variables.

Theorem 3 *The feasible set of the optimization model composed by (4a) - (4e) and (5) - (6) is equivalent to the feasible set of the optimization model composed by (4a) - (4e) and the constraints (7)-(8)*

$$y_i^k - y_j^k \leq 2 - x_i - x_j \quad \forall k \in \{1, ..., K\};\ \forall i, j \in \{1, \ldots, N\} : i > j \text{ and } \sigma^k(i) = \sigma^k(j) \tag{7}$$

$$-y_i^k + y_j^k \leq 2 - x_i - x_j \quad \forall k \in \{1, ..., K\};\ \forall i, j \in \{1, \ldots, N\} : i > j \text{ and } \sigma^k(i) = \sigma^k(j) \tag{8}$$

Proof: The equivalence is easiest to show by considering the possible values of binary variables x_i and x_j in a feasible solution. If at least one of the two variables has value 0, the absolute value of $y_i^k - y_j^k$ is neither further constrained by (5) nor by (7)-(8). If both variables have value 1, both (5) and (7)-(8) force y_i^k and y_j^k to take equal values. Therefore, both sets of constraints have the same impact on the y variables, resulting in equivalent feasible sets. ∎

One may therefore directly optimize the assortment based on partially-ranked consumer behaviors without introducing new variables. While the structure of problem (4) forced the continuous $\boldsymbol{y}$ variables to take binary values, adding constraints (7) and (8) breaks this structural property, allowing the variables to take any continuous value between 0 and 1. As outlined above, these constraints have an intuitive interpretation. They force y_i^k and y_j^k to take the same value if both products i and j are part of the assortment and have equal rank in customer behavior k. In this case, if none of the higher ranked products is available in the assortment, each of y_i^k for the indifferent products will take value $\frac{1}{|I(\sigma) \cap \mathcal{S}|}$, where $\mathcal{S}$ is the assortment defined by variables $\boldsymbol{x}$. Even though there is a quadratic number of constraints, those are only on the size of the indifference sets. If the indifference sets are large, one may generate those constraints on the fly, adding only those constraints that are violated in the linear programming solution in each of the Branch-and-Bound nodes.

4 Computational Results

We now focus on empirical experiments performed with the proposed non-parametric choice model. In Section 4.1, we will evaluate the performance of the choice model and the assortment optimization algorithms on synthetic data. We are particularly interested in evaluating how well those models perform in terms of scalability and ability of learning the choice model. In Section 4.2, we show

the usefulness of our approach in practice by performing an analysis on industrial data sets from the clothes retail sector.

All computational experiments have been carried out on a single Intel(R) Xeon(R) X5675 3.07GHz processor, limited to 48 GByte of memory. The algorithms have been coded in Python version 3.6.1. Mathematical models have been solved using the MIP solver of Gurobi version 7.0.2.

4.1 Numerical Results on Synthetic Data

Data Generation. We generate sales and assortment data according to a ground-truth (GT) model. The data generated according to this GT model will be used to evaluate the performance of the non-parametric choice models discussed so far and the corresponding algorithms proposed in the previous sections. We choose as GT model a Mixed Multinomial Logit model with T classes of customers. The probability distribution among the classes are drawn from the T-dimensional simplex p_t (therefore, $\sum_t p_t = 1$ and $p_t \geq 0 \ \forall t$). Each customer class t associates an utility $u_{t,i}$ with a product i and selects product i from assortment S with the following probability

$$\mathbb{P}(i|S, \text{class}{=}t) = \frac{e^{u_{t,i}}}{e^{u_{t,0}} + \sum_{j \in S} e^{u_{t,j}}}.$$

The overall probability that a random customer chooses a product i is therefore given by

$$\mathbb{P}(i|S) = \sum_{t=1}^{T} p_t \frac{e^{u_{t,i}}}{e^{u_{t,0}} + \sum_{j \in S} e^{u_{t,j}}}.$$

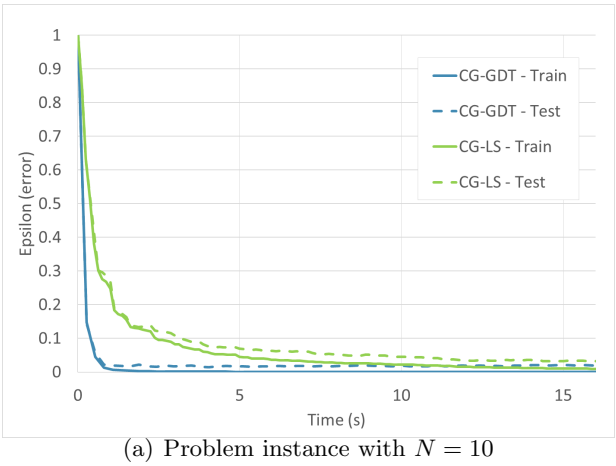
The utilities for each customer class are generated as proposed by Bertsimas and Mišic (2016). Specifically, we generate a matrix q of the same dimension as u uniformly distributed on $[0, 1]$. If not stated otherwise, 4 of the $N + 1$ products from $\mathcal{N} \cup \{0\}$ are randomly selected for each customer class t. Those products are assumed to have high utilities for customer class t, computed as $u_{t,i} = 10 * q_{t,i}$. The utilities for the remaining $N - 4$ products are set to $u_{t,i} = 0.1 * q_{t,i}$. The training and test sets for the choice models consist each of M assortments. In all experiments, M has been set to 20 to reflect a context where the number of historical observations is limited. Assortments densities r are set to 0.5, i.e., each assortment contains $N/2$ products. Utilities $u_{t,i}$ are translated to a vector of sales probabilities $v_{i,m} = \mathbb{P}(i|S_m)$ for each product in each assortment. Sales transactions are then randomly generated according to the sales probabilities. Recall that the sales vector $\boldsymbol{v}$ is indexed by tuples (i, m). Therefore, the choice matrix $A^k_{i,m}$ has two dimensions.

4.1.1 Training the Choice Model

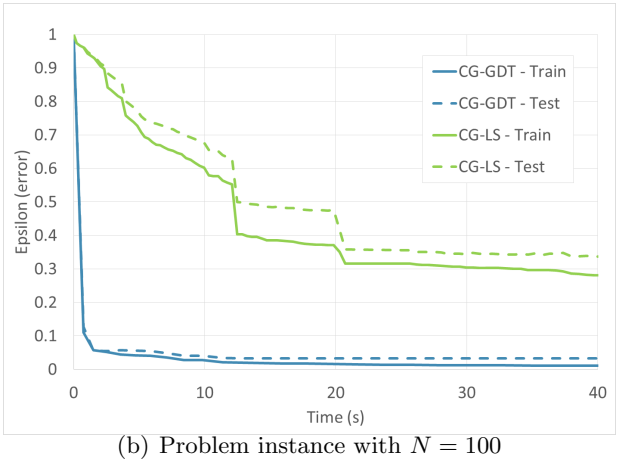
In the following, we will investigate how well the proposed choice model can be trained using the Growing Decision Tree algorithm when compared to classical approaches. We then analyze both sparsity and concision of the generated choice models.

Convergence and scalability. The choice model is trained by iteratively generating new customer preference lists via column generation and solving master problem which minimizes the estimation error based on the training data. Let $\boldsymbol{A}_{tr}$ and $\boldsymbol{v}_{tr}$ be the choice matrix and the sales probabilities for the M assortments of the training set. At each iteration, we compute the current training error as $\epsilon_{tr} = \dfrac{|\boldsymbol{A}_{tr}\boldsymbol{\lambda} - \boldsymbol{v}_{tr}|}{2M}$. The test error is computed in the same way, but using choice matrix $\boldsymbol{A}_{te}$ and sales vector $\boldsymbol{v}_{te}$ based on the M assortments of the test data. The training algorithm terminates once the training error ϵ_{tr} is smaller or equal to threshold ϵ_0, which is set to 0.01, if not stated otherwise.

Figure 3 (a) and (b) show the evolution of the training and test errors for problem instances with $N = 10$ and $N = 100$, respectively. The curves are plotted for two training algorithms: CG-GDT refers to the column generation approach that generates new columns based on the Growing Decision Tree. CG-LS refers to a column generation approach based on the work of Bertsimas and Mišic (2016), using a local search to find new fully-ranked preference lists. With 10 products (see Figure 3 (a)), we notice that the training error ϵ_{tr} is below 1% in less than a second using the CG-GDT, whereas the CG-LS takes about 15 seconds to achieve the same accuracy. In both cases, the test errors (dotted lines) are slightly higher than the training errors. However, they are strictly decreasing, which illustrates that the approaches are not affected by overfitting. For a larger instance with $N = 100$ products (see Figure 3 (b)), we notice that the CG-GDT quickly achieves a small training error, whereas the CG-LS requires more time to find the right columns that decrease the estimation errors. In particular, in the plotted time period, the algorithm has not been able to reduce the error below 37%, while the CG-GDT choice model achieves an error of $1,5\%$ in 25 seconds.



(a) Problem instance with $N = 10$



(b) Problem instance with $N = 100$

Figure 3: Learning curves (training and test error) for CG-GDT and CG-LS on example problem instances.

To investigate the scalability of the two training approaches, we generated 10 random instances for each $N \in \{30, 50, 100, 250, 500, 1000\}$. Both approaches, the CG-GDT and CG-LS, have been limited to 12 hours computing time. The average results over the 10 instances are reported in

Table 1 for each of the two approaches and for different problem sizes N. The table reports the final training error, the computing times, the number of iterations, and the final number K of generated preference lists with non-negative probabilities λ_k. Problem instances that hit the 48 Gbyte memory limit have not been considered in the average values. The number of those instances is reported in column "# inst oom". The CG-GDT has successfully trained the choice model to the required threshold of $\epsilon_0 = 0.01$ (which refers to a training error of 0.4 when $M = 20$) for all instances within the given time and memory limits. In contrast, the CG-LS hits those limitations for instances with 1,000 products, and in some cases for 500 products. Generally, the CG-GDT is more scalable than the CG-LS, and, given the smaller number of iterations necessary to converge, the former produces significantly sparser choice models (in the sense that the number of customer behaviors required to explain the sales is smaller; see Farias et al. (2013)). For the CG-GDT, one observes that the number of iterations is relatively low for large N. This is explained by the fact that the indifference sets have a larger explanatory power in those instances, which will be further explored in the following paragraph.

Concision and explanatory power of the indifference sets. The previous results have suggested that the CG-GDT approach scales well even when the number of considered products N is relatively high. We now analyze the choice model generated by this approach. As shown above, the CG-GDT tends to produce significantly sparser choice models, i.e., it requires less preference lists to explain the observed sales. This is a benefit in practice. Managers may further be interested in obtaining concise preference lists, i.e., those that only require few strictly ranked products. In Section 2, Propositions 1 and 2 have suggested that only few ranks in the preference lists are important to explain the transaction data. We will next verify empirically whether the computational results support these statements.

We will first explore how the training accuracy threshold ϵ_0 impacts the produced choice models. Table 2 shows several properties for the choice models generated by CG-GDT for different training accuracies $\epsilon_0 \in \{0.1, 0.01, 0.001\}$ and problem sizes N. The results are averaged over 10 random instances and include the average size K of the choice models and the number of strictly ranked

	CG-GDT				CG-LS				
N	Train. error	time (sec)	# iter	K	# inst. oom	Train. error	time (sec)	# iter	K
30	0.37	2.3	9.2	105.6	0	0.39	22.5	392.0	223.8
50	0.38	6.0	10.3	104.7	0	0.40	57.3	603.2	370.1
100	0.39	29.7	15.4	127.3	0	0.40	269.8	1,070.7	721.3
250	0.39	321.8	21.0	213.3	0	0.40	5,204.8	2,492.9	1,788.7
500	0.38	2,341.5	19.4	416.6	1	0.40	49,615.3	4,555.0	3,484.2
1000	0.33	5,511.2	7.0	850.2	10	-	-	-	-
all (avg)	0.38	1,368.7	13.7	303.0	11	0.40	10,459.6	1,795.6	1,295.3

Table 1: Learning performance for CG-GDT and CG-LS algorithms with $M = 20$ and $\epsilon_0 = 0.01$ (averaged over 10 random instances)

products (average and maximum number). The last two columns reveal information about the explanatory power of the indifference sets, i.e., the percentage of sales that are explained by indifference sets. The exact percentage for a given choice model can be computed as $\sum_{k \in \mathcal{K}} \lambda_k \cdot \frac{numInd_k}{M}$, where $numInd_k$ is the number of (k, m) tuples in which at least one product from an indifference set has a value greater than 0. In words, it is the weighted ratio between the number of assortments in which a product from the indifference set has been sold and the total number of sales (which equals the total number of assortments M, if the selection of 0 is considered a sale). This percentage is indicated in column "exact comp.". The last column, "theor. est.", is linked to the theoretical approximation of the indifference percentage in Proposition 2. In this proposition, the exact percentage is computed for an average assortment, explicitly using the different λ_k values for each preference list k. Since we are not dealing with average assortments, taking all λ_k into account does not make the result more informative for our case of a specific assortment. We therefore report in column "theor. est." the value given by the simplified formula $(1 - r)^{avgRanked}$, where $avgRanked$ is the average value reported in column "# strictly ranked products avg" of the same line (with $r = 0.5$).

The results in Table 2 indicate that, as the training is more accurate and ϵ_0 is decreased, the number of required GDT iterations and the size of the final choice model increase. The number of strictly ranked items also slightly increases. However, it generally remains quite low and never exceeds more then 6 strictly ranked products in any of the generated preference lists, which is only a fraction of the total number of products (i.e., up to 1000). The proportion of sales explained by products in the indifference set steadily decreases as the training accuracy is increased (i.e., ϵ_0 is decreased). This illustrates the high explanatory power of the first few ranked products if the choice model is well chosen. For example, with $\epsilon_0 = 0.001$ and $N = 250$ products, all preference lists contain 4 or less strictly ranked products, which explain 64.35 % of the sales transactions, while only 35.65 % of the transactions are explained by the remaining 246 products that are not strictly ranked. While classical approaches using fully-ranked preference lists will always contain N strictly ranked products, the proposed approach based on partially-ranked preference lists allows store managers to gain valuable insights from a small list of products that have a fairly high explanatory power. Interestingly, the theoretical estimation of this percentage is quite close to that practically computed, which confirms our theoretical findings in Proposition 2.

Recall that in the experiments above, the ground-truth model to generate the problem instances contains, for each customer class, four products with high utilities. We now explore how the number of products with high utilities in the underlying ground-truth model impacts the final choice model. Table 3 summarizes the same results as in the previous table for $\epsilon_0 = 0.01$, but based on ground-truth models that assume that each customer class has exactly 1, 4, 10 and 20 products with high utilities. As the number of products with high utilities increases, the algorithm requires more iterations to find a choice model that fits the transaction data accurately. However, the number of preference lists with non-negative probabilities remains similar in all cases. The number of strictly ranked products also remains surprisingly stable, indicating that a final accurate choice model is

ϵ_0	N	# iter	K	# strictly ranked products		% explained by indifference sets	
				avg	max	exact comp.	theor. est.
0.1	30	1.0	34.6	1.36	2	36.36	39.05
0.1	50	1.0	41.3	1.07	2	45.03	47.78
0.1	100	1.0	82.5	1.01	2	48.70	49.75
0.1	250	1.0	219.4	1.00	2	49.98	49.95
0.1	500	1.0	444.3	1.00	2	49.67	49.99
0.1	1000	1.0	913.6	1.00	1	49.74	50.00
0.1	all	1.0	289.3	1.07	2	46.58	47.57
0.01	30	10.2	105.6	2.24	4	20.28	21.22
0.01	50	11.3	104.7	1.84	4	29.69	27.85
0.01	100	16.4	127.3	1.55	3	36.35	34.17
0.01	250	22.0	213.3	1.22	3	44.76	43.07
0.01	500	20.4	416.6	1.07	3	47.78	47.69
0.01	1000	8.8	836.2	1.03	2	48.54	48.98
0.01	all	14.9	300.6	1.49	4	37.90	35.59
0.001	30	29.6	189.3	2.74	6	15.03	14.93
0.001	50	31.0	190.9	2.35	5	19.91	19.55
0.001	100	49.6	216.7	1.91	4	29.41	26.56
0.001	250	115.5	287.9	1.69	4	35.65	31.07
0.001	500	149.7	438.4	1.37	3	42.00	38.61
0.001	1000	53.3	743.9	1.10	2	46.51	46.51
0.001	all	71.5	344.5	1.86	6	31.42	27.50

Table 2: Properties of choice models generated by CG-GDT (average values over 10 random instances) with different training error thresholds ϵ_0 ($M = 20$).

not more complex, but only more difficult to find. Finally, one observes that such more refined choice models also reduce the percentage of sales transactions that are explained by the products in the indifference sets. As before, the theoretical estimation of this percentage is close to that practically computed.

4.1.2 Assortment Optimization

In the previous section, we have shown that the partially-ranked choice model can be accurately learned in reasonable computing times even in contexts with large numbers of products. We now explore the scalability of the mathematical models to optimize assortments once the choice model has been learned. We will focus on three different approaches. The first two approaches learn a partially-ranked choice model using the CG-GDT algorithm. The first approach performs subsequent assortment optimization adding the indifference inequalities (7) and (8) via branch-and-cut (referred to as AO-B&C). The second approach performs boosting to create several fully-ranked preference lists at random and then optimizes via the classical MIP (4) based on fully-ranked lists (referred to as AO-Boost). Finally, the third approach is based on the CG-LS with the classical MIP (4) (referred to as AO-Compl).

In the following, we will first explore how well one may tune the boosting approach AO-Boost to approximate the exact approach AO-B&C. Then, we focus on a direct comparison of the three approaches.

Boosting. Even though we can add the indifference constraints (7) and (8) via branch-and-cut to the assortment optimization MIP (4) to directly operate on a partially-ranked choice model $\sigma = (P(\sigma), I(\sigma))$, we may attempt to complete the strictly ranked products by imposing a strict order on the products in the indifference set $I(\sigma)$. Creating several of those fully-ranked preference lists at random is called *boosting*.

We define two parameters to control the total number of fully-ranked lists and to assure that a preference list σ_k with high probability λ_k yields more fully-ranked lists than a σ_k with low probability λ_k. We define n_{min} as the minimum number of lists generated for each of the original preference lists. We also define τ as a scale parameter to control the magnitude of lists generated in proportion to the value of λ_k. For each partially-ranked preference list σ_k, we generate $n_{min}-1+\tau\lambda_k$ lists in which the products in the indifference set are ordered at random.

Table 4 compares the sizes K of the generated choice models and the average revenues of the optimized assortments for the exact approach AO-B&C and the boosting approach AO-Boost with different values for parameter τ (with $n_{min} = 3$). In AO-B&C, the indifference constraints are added via user callbacks by adding the first 2,500 violated constraints at each callback. Revenues are reported as average values of the expected revenue, which refers to the objective function value of the optimization problem, and as the revenue as evaluated by the ground-truth model. For both revenue types, the table reports the average deviation of the revenue given by the boosting approach from the revenue given by the exact AO-B&C approach, as well as the corresponding

ϵ_0	# prod. w/ high utility	N	# iter	K	# strictly ranked products		% explained by indifference sets	
					avg	max	exact comp.	theor. est.
0.01	1	30	5.2	80.7	1.79	4	33.52	28.97
0.01	1	50	5.4	76.2	1.54	3	38.53	34.45
0.01	1	100	7.4	103.2	1.32	3	40.69	40.06
0.01	1	250	7.2	221.1	1.06	2	48.27	47.92
0.01	1	500	12.3	436.2	1.05	2	48.12	48.34
0.01	1	1000	6.5	879.2	1.03	2	48.55	49.09
0.01	1	avg/max all	7.3	299.4	1.30	4	42.95	40.70
0.01	4	30	10.2	105.6	2.24	4	20.28	21.22
0.01	4	50	11.3	104.7	1.84	4	29.77	27.85
0.01	4	100	16.4	127.3	1.55	3	36.35	34.17
0.01	4	250	22.0	213.3	1.22	3	44.76	43.07
0.01	4	500	20.4	416.6	1.07	3	47.78	47.69
0.01	4	1000	7.9	834.7	1.03	2	48.62	48.94
0.01	4	avg/max all	14.7	300.4	1.49	4	37.93	35.58
0.01	10	30	18.4	120.3	2.50	6	16.58	17.63
0.01	10	50	17.3	119.7	2.13	4	21.10	22.87
0.01	10	100	23.4	136.3	1.69	4	31.79	30.92
0.01	10	250	36.0	227.0	1.31	3	42.72	40.36
0.01	10	500	48.3	392.1	1.18	3	45.49	44.21
0.01	10	1000	23.5	823.8	1.04	2	48.77	48.66
0.01	10	avg/max all	27.8	303.2	1.64	6	34.41	32.04
0.01	20	30	9.3	111.0	2.09	4	24.40	23.44
0.01	20	50	16.0	119.4	2.11	4	22.66	23.14
0.01	20	100	28.4	126.3	1.87	4	24.60	27.27
0.01	20	250	47.7	214.3	1.43	3	39.08	37.12
0.01	20	500	71.6	377.8	1.26	3	43.82	41.68
0.01	20	1000	36.8	749.0	1.08	2	47.71	47.33
0.01	20	avg/max all	35.0	283.0	1.64	4	33.71	32.05

Table 3: Properties of choice models generated by CG-GDT (average values over 10 random instances) with different numbers of products with high utilities in ground-truth model ($M = 20$, $\epsilon_0 = 0.01$).

standard deviation.

			Expected revenu			GT revenu		
				Gap % from			Gap % from	
		time	avg	AO-B&C		avg	AO-B&C	
	K	(sec)	rev	avg	std-dev	rev	avg	std-dev
AO-B&C	125.7	45.1	86.18	-	-	86.31	-	-
$\tau = 10$	376.6	27.0	86.58	3.38	10.06	84.50	3.18	10.10
$\tau = 50$	390.0	30.8	86.21	2.55	2.71	84.43	2.25	2.55
$\tau = 100$	430.0	35.1	86.06	2.27	2.55	84.55	2.02	2.64
$\tau = 500$	817.0	97.5	85.57	1.69	2.92	85.07	1.43	2.45
$\tau = 1,000$	1,315.4	224.9	85.42	1.57	2.85	85.37	1.25	2.48
$\tau = 5,000$	5,315.4	4,394.4	85.21	1.30	3.08	85.47	0.97	2.44
$\tau = 10,000$	10,314.5	16,860.7	85.16	1.24	3.09	85.49	0.99	2.40

Table 4: Assortment optimization approximation via boosting compared to AO-B&C algorithm (averaged over 100 random instances; $N = 100$, $M - 20$ and $r = 0.3$).

As τ increases, both the expected and the GT revenus provided by AO-Boost get closer to the exact revenus as given by AO-B&C. However, the number of fully-ranked preference lists generated in the final choice model quickly grows (and, as a consequence, the computing times as well). The resulting optimization models are too difficult to solve and not competitive with the exact branch-and-cut approach AO-B&C.

Scalability. As a final study, we now investigate how well the three assortment optimization approaches scale to large number of products. Table 5 reports, for each approach, the average size K of the choice model, the average computing times to solve the optimization model and the average revenue as computed by the ground-truth model. If applicable, we also report the number of problem instances which have run out of memory ("oom").

	CG-GDT - AO B&C				CG-GDT - AO-Boost				CG-LS - AO-Compl		
		time	GT	#		time	GT	#		time	GT
N	K	(min)	revenu	oom	K	(min)	revenu	oom	K	(min)	revenu
30	109.9	0.1	74.5	0	386.8	0.0	74.2	0	220.0	0.0	73.6
50	113.5	0.1	82.5	0	397.9	0.1	81.9	0	379.8	0.1	81.9
100	117.8	0.8	88.8	0	407.7	0.6	86.0	0	722.0	2.0	86.3
250	211.0	7.3	90.4	0	655.5	9.5	88.9	0	1,813.1	141.3	89.7
500	438.1	113.1	94.5	0	1,321.7	249.4	92.9	10	-	-	-
1000	897.4	669.9	95.0	10	-	-	-	10	-	-	-
all (avg)	314.6	131.9	87.6	10	633.9	51.9	84.8	20	783.7	35.8	82.9

Table 5: Assortment optimization results for choice models generated by CG-GDT and CG-LS (averaged over 10 random instances).

As the number of products N increases, all choice models become more complex. The CG-LS optimization approach handles only up to 250 products within the given memory limits. The CG-

GDT approach with boosting handles problem instances with up to 500 products. Finally, given the moderate size of the choice models, the AO-B&C approach handles solves all problem instances with up to 1,000 products. Remarkably, this approach does not only result in shorter computing times, but also yields higher revenues as computed by the ground-truth model.

We conclude this section noting that the recent work of Bertsimas and Mišic (2017) also conducted numerical experiments on the original formulation (4) for fully-ranked lists. Considering that the more recent version of the MIP solver used in Bertsimas and Mišic (2017) results in slightly faster computations, the computational results are consistent with our findings. These authors further propose a Benders decomposition implementation, which could also be adapted to our formulation. It has to be noted, however, that the assortment optimization MIP itself is not the only crucial ingredient of the overall approach. Learning the choice model correctly and accurately for large-scale problems is, as has been shown, computationally challenging and crucial to obtain meaningful assortment optimization models and, in this concern, we believe the partially-ranked choice model provides an efficient option.

4.2 Case Study on Industrial Retail Data

We will now discuss an industrial case study based on real world data from a North-american clothes retailer. Two anonymized data sets have been obtained from our industrial partner JDA Labs (2017): one for shoe stores and one for shirt stores. In the following, we will discuss the data sets and preprocessing. Then, we will explore the converge performance of the different approaches to train the choice models on one data set, and the outcomes of the assortment optimization under capacity constraints on the other data set.

4.2.1 Data description and preprocessing

Both the shoe and the shirt data sets include assortment data, transaction data, characteristics of all available products, and characteristics of the stores from August 2014 to July 2015. Assortment information is provided for each day and each store (with information such as location, climate, price category). Sales transaction data contains product IDs, sold quantities, sales time-stamps and store ID. Products in the shoe dataset have characteristics given as categorical values (class, sub-class, brand, material, color) and continuous values (average price). Products in the shirt data set contain additional information (lifestyle, pattern, fit, sleeve length, fashion).

Based on those information, we define an assortment as the set of products offered in a particular store throughout one calendar week in a particular year. For each assortment, we link the corresponding sales transactions and convert those into the vector of sales probabilities $\boldsymbol{v}$, representing the probability of selling a certain product in a given assortment. The following describes the entire process of data preparation. For more details, we refer to the Master thesis of Palmer (2017).

Store clustering. Assuming that stores in neighborhoods with similar characteristics meet the needs of similar customer types, we need to learn separately for groups of similar stores. As a result of the frequent discussions with JDA Labs (2017), which provided the data, we clustered the stores according to four store features: location (state and city), climate (4 different categorical values), price band (low, medium and high), and percentage of sales in each sub-category. The most popular algorithm for non-supervised clustering is k-means, which requires the data values to be continuous. However, our features contain both categorical and continuous values. We therefore use an extension of k-means that can handle mixed categorical and continuous data (Huang, 1997).

Data preprocessing and no-purchase estimation. We arbitrarily selected a cluster from the shoes data set that contains 10 stores and has fairly high sales. For each of these stores, we use data during 10 consecutive weeks from Autumn 2014, which we consider a good trade-off to have sufficient data, while not risking that store assortments changed much due to seasonal fashion. We considered each week of store data as a proper assortment, resulting in a total of 100 assortments. To ensure that the historical data is statistically meaningful, we only considered products that have been sold at least 10 times, resulting in a final total of 192 different products.

Given that we did not have any information about how many customers left the store without purchase, we estimated the no-purchase probabilities assuming that assortments with many sales have lower no-purchase probabilities and assortments with few sales have higher no-purchase probabilities. We further assume that the no-purchase probabilities of the different assortments lie between 10% and 30%. Let $v_{0,m}$ be the no-purchase probability in assortment m. Let $sales_m$ be the number of sales observed in assortment m. Let $sales_{MIN} = \min_{m\in\mathcal{M}}\{sales_m\}$ and $sales_{MAX} = \max_{m\in\mathcal{M}}\{sales_m\}$. We then computed the no-purchase probability for a store m as a linear interpolation between 0.1 and 0.3 according to the number of observed sales: $v_{0,m} = 0.1 + 0.2 \cdot \frac{sales_m}{sales_{MAX} - sales_{MIN}}$. Let $sales_{i,m}$ denote the number of sales of product i in assortment m. We then compute the sales probability for any other product i as $v_{i,m} = (1 - v_{0,m}) \cdot \frac{sales_{i,m}}{sales_m}$, which is the corresponding sales proportion of product i taking into consideration the probability for the no-purchase option.

4.2.2 Computational Results

Convergence of training the choice model. We use data from 10 stores during 10 consecutive weeks from Autumn 2014 from the shoes data set, resulting in 100 assortments. These assortments hold a total of 192 different products. Figure 4 plots the convergence curves for the CG-GDT and CG-LS training approaches. We notice that the optimal training error is not 0 (as it is the case for synthetic data), because real data is noisy, or even contradictory.[1] The training error quickly converges to 41% using the GDT choice model, while CG-LS remains at 48% after 500 seconds.

[1] As an example, consider two assortments $S_1 = \{1, 2, 3\}$ and $S_2 = \{1, 2, 4\}$. We may have observed only sales of product 1 in assortment S_1, and only sales of product 2 in assortment S_2. In this case, a ranking-based choice model cannot perfectly fit the sales transactions for both assortments.

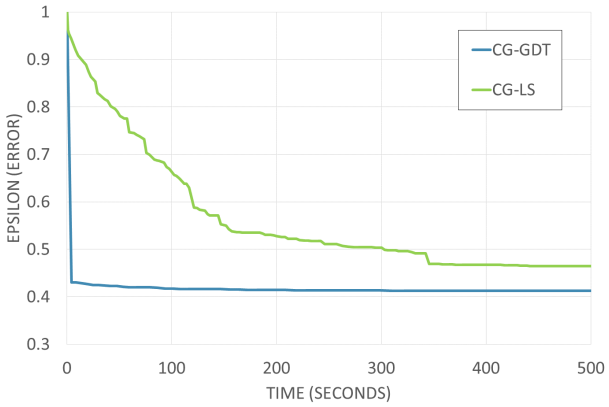


Figure 4: Learning curves for CG-GDT and CG-LS on industrial shoe data with 192 products.

Assortment optimization with capacity constraints. In practice, retailers are typically restraint by space limitations and carefully need to decide which products to offer to respect the available space capacity. As already observed by Bertsimas and Mišic (2016), we may add the following constraint to the MIP formulation (4) to account for capacity limitations:

$$\sum_{i=1}^{n} x_i \leq U.$$

Experiments have been carried out on the shirts data sets on 7 stores and their corresponding sales data for the months of April to July 2015. Given that the shirt data set contains significantly more sales than the shoe data set, each assortment has been defined by the transactions occurred during one day. This also enabled us to further split the assortments into a training and a test set. We excluded products that were sold less than 10 times in the entire data set, considering that such data occurrence was not sufficient to perform accurate predictions. In total, the assortments contained 196 different products.

We first trained via CG-GDT to near-optimality. Then, we solved the assortment optimization model with the capacity constraints above. The optimal assortment contained 91 of the 196 products. We were particularly interested to explore how smaller store capacities would impact the revenue. Since we deal with real data, we did not have access to a ground-truth model to specify the revenue of an assortment that has not been observed in the past. We will therefore assume that customers behave as given by the final choice model and compute each of the predicted revenues for the assortments in the test set. We denote by R^*_{test} the highest revenue as computed by the choice model among all assortments in the test set, i.e., assuming that the choice model has been accurately trained, none of the past assortments in the test set achieves a better revenue than R^*_{test}. The potential revenue increase when using a new assortment can then be computed as

$$\textit{Rev. Incr.} = \frac{R^*_{GDT} - R^*_{test}}{R^*_{test}},$$

where R^*_{GDT} refers to the predicted revenue of the optimized assortment:

We then solved the assortment optimization problem, limiting the capacities to $U \in \{25, 40, 50, 65, 100\}$. The optimal solution with 91 products is found only with $U = 100$. For all other capacities, the optimal assortments have been found to always contain the maximum number of products U. However, with smaller capacities, the revenue increase is expected to be attenuated. The potential revenue increase for each value of U is plotted in Figure 5. While the potential revenue increase is predicted to be as high as 45%, the potential increase with only 25 products is predicted to 25%, which is still a surprisingly significant increase.

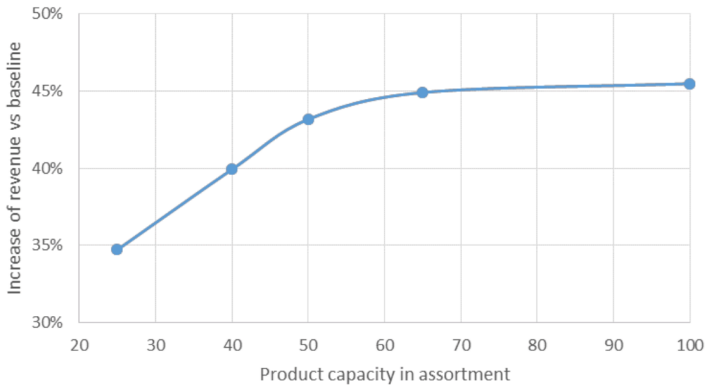


Figure 5: Impact of the capacity constraint on the predicted revenue increase

5 Conclusion

In this work, we have focused on non-parametric rank-based choice models for assortment optimization. Those choice models have several advantages. Mainly, they can be estimated in a purely data-driven manner without relying on previous knowledge on the market structure. They also tend to be insensitive to overfitting. Our work proposes a new methodology to estimate those choice models, which scales to large number of products. In particular, we propose to represent customer behaviors not by fully ordered preference lists of all products, but only a subset of them. This is a realistic setting, which directly exploits the fact that products with low ranks have little explanatory power and impact in the buying behavior. Further, we show that any partially-ranked choice model can be transformed into an equivalent fully-ranked choice model, and vice-versa. However, the equivalent fully-ranked choice model contains a number of preference lists that is factorially large in the number of products in the indifference sets, making it intractable for assortment optimization. In contrast, the partial representation of the strictly ranked products enables us to efficiently train the choice model by gradually expanding a tree, in which each of the nodes represents partial lists of strictly ranked products. On this particular structure, new preference lists can be found efficiently via column generation. We finally present new inequalities to adapt the classical assortment optimization model to our partially-ranked choice models. Extensive compu-

tational experiments have shown that instances with up to 1,000 products can be efficiently trained and assortments can be optimized in quite low computing times, increasing by one order of magnitude the capabilities of previous approaches to learn the choice model. Given that training the partially-ranked choice model by means of a growing tree and column generation has been proven to be very efficient in the case of assortment optimization, it may be a promising avenue to explore it any other context in which discrete choice models are central.

Acknowledgements. The authors are thankful to JDA Labs, in particular to Marie-Claude Côté, for providing the industrial data sets and for their support throughout this research. The work of the first author was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada. The work of the third author was mostly conducted when he was a Research Master student at Polytechnique Montréal in the Canada Excellence Research Chair "Data Science for Real-time Decision-making", whose support is strongly acknowledged.

References

Aouad, A., V. Farias, R. Levi, D. Segev. 2015. The Approximability of Assortment Optimization under Ranking Preferences. *Working paper, Massachusetts Institute of Technology and University of Haifa* .

Arrow, Kenneth Joseph. 1951. *Social Choice and Individual Values*. 12, John Wiley & Sons.

Belloni, A., R. Freund, M. Selove, D. Simester. 2008. Optimizing product line designs: Efficient methods and comparisons. *Management Science* **54**(9) 1544–1552.

Ben-Akiva, M. E. 1973. Structure of Travel Demand Models. Ph.D. thesis, Massachusetts Institute of Technology.

Bertsimas, D., V. Mišic. 2016. Data-driven assortment optimization. *Working paper, Massachusetts Institute of Technology* .

Bertsimas, D., V. Mišic. 2017. Exact first-choice product line optimization. *Working paper, Massachusetts Institute of Technology and University of California* .

Farias, V. F., S. Jagabathula, D. Shah. 2013. A nonparametric approach to modeling choice with limited data. *Management Science* **59**(2) 305–322.

Ho-Nguyen, N., F. Kilinc-Karzan. 2017. Dynamic Data-Driven Estimation of Non-Parametric Choice Models. *Working paper, Carnegie Mellon University* .

Honhon, D., S. Jonnalagedda, X. A. Pan. 2012. Optimal Algorithms for Assortment Selection Under Ranking-Based Consumer Choice Models. *Manufacturing & Service Operations Management* **14**(2) 279–289.

Huang, Z. 1997. Clustering large data sets with mixed numeric and categorical values. *In The First Pacific-Asia Conference on Knowledge Discovery and Data Mining*. 21–34.

Jagabathula, S. 2011. Nonparametric Choice Modeling: Applications to Operations Management. Ph.D. thesis, Massachusetts Institute of Technology.

JDA Labs. 2017. `https://jda.com/innovation/jda-labs`. (Accessed: 2017-09-07).

Kök, A. G., Ma. L. Fisher, R. Vaidyanathan. 2008. Assortment planning: Review of literature and industry practice. *Retail supply chain management*. Springer, 99–153.

Mahajan, S., G. J. van Ryzin. 1999. Retail inventories and consumer choice. *Quantitative models for supply chain management*. Springer, 491–551.

Palmer, Hugo. 2017. Large-scale Assortment Optimization. Master's thesis, Polytechnique Montréal.

Vulcano, G., G. Van Ryzin. 2017. Technical Note - An expectation-maximization method to estimate a rank-based choice model of demand. *Operations Research* **65**(2) 396–407.