On the Relevance of Irrelevant Alternatives

Austin R. Benson*
Stanford University
Stanford, CA
arbenson@stanford.edu

Ravi Kumar
Google Inc.
Mountain View, CA
ravi.k53@gmail.com

Andrew Tomkins
Google Inc.
Mountain View, CA
atomkins@gmail.com

ABSTRACT

Multinomial logistic regression is a powerful tool to model choice from a finite set of alternatives, but it comes with an underlying model assumption called the *independence of irrelevant alternatives*, stating that any item added to the set of choices will decrease all other items' likelihood by an equal fraction. We perform statistical tests of this assumption across a variety of datasets and give results showing how often it is violated.

When this axiom is violated, choice theorists will often invoke a richer model known as nested logistic regression, in which information about competition among items is encoded in a tree structure known as a *nest*. However, to our knowledge there are no known algorithms to induce the correct nest structure. We present the first such algorithm, which runs in quadratic time under an oracle model, and we pair it with a matching lower bound.

We then perform experiments on synthetic and real datasets to validate the algorithm, and show that nested logit over learned nests outperforms traditional multinomial regression.

Finally, in addition to automatically learning nests, we show how nests may be constructed by hand to test hypotheses about the data, and evaluated by their explanatory power.

Keywords:. discrete choice; independence of irrelevant alternatives; logit

1. INTRODUCTION

In this paper, we consider the problem of *discrete choice*, in which a user must select one element from a set of non-overlapping and exhaustive alternatives. The element might be a car to buy, a flight to take, or an apartment to rent. Our task is to estimate the likelihood of each choice given a particular slate of alternatives.

The rich literature in this domain begins with Luce's Axiom of Choice [19], also referred to as the *Independence of Irrelevant Al-*

.nIIA-j/T1_4 8.966 Tf[13.78 0 Td[(dor)2260(sthor.)-530(,Tis)-360(aniom)-360(statie)-360(ahat)-2060the reflaive rikel-]TJ-63.854 -10.461 Td[(wihood)-23

logit is employed. When IIA does not hold, modelers focus primarily on introducing hierarchy to relax IIA via a generalization of multinomial logit known as *nested logit*.

To begin with an example, consider the scenario above in which a new liberal candidate competes with an existing liberal candidate, modifying the relative likelihood of the original liberal and conservative candidates. In a hierarchical model, the universe of candidates would be represented in a depth-two tree split at the top level into a "nest" of conservatives and another nest of liberals. In such a model, a user would first determine whether to vote liberal or conservative, and then draw a candidate from the selected nest according to a standard multinomial.[2] This hierarchical structure would correctly explain the change in relative likelihood when another liberal candidate is added to the slate, as the amount of probability assigned to the liberal nest is fixed independent of the number of alternatives available within that nest.

Researchers applying nested logit will typically hand-build a "natural" tree, and then employ statistical techniques to determine how well this tree models the data. However, in modern large-scale datasets, there are often sufficiently large numbers of choices available, that hypothesizing one particular hierarchical relationship is simply infeasible. In such a situation, data miners would prefer to induce the structure of the tree directly from data. However, to our surprise, despite the broad use of nested logit, we have not discovered any work on the problem of nested logit tree induction.

In this paper, we begin the study of this problem. We consider data that has been drawn from a nested logit model, and develop a quadratic-time algorithm to recover the nest structure using a natural oracle model that provides information about the underlying tree structure. We show a matching quadratic lower bound, so that under the particular oracle we study, no further improvements in asymptotic complexity are possible.

With this algorithm in hand, we may then proceed to analysis of data. We show our algorithm successfully recovers existing nest structure using a set of synthetic evaluations. We then employ our algorithm to model real-world datasets. Our algorithm in many cases unearths trees that in post-hoc evaluation make sense, and increase likelihood of the data compared to standard multinomial regression. However, we also discover situations in which nested logit performs worse than multinomial, and we describe some plausible causes for this phenomenon.

To summarize, our contributions are the following.

(i) We study several choice datasets and develop statistical tests that operate well for large-scale data with highly heterogeneous choice sets. Using these tools, we study the prevalence of IIA in real online datasets. We show that, to our surprise, there exist complex datasets whose elements seem likely to compete preferentially within particular subsets, for which IIA is an extremely good model. However, we also show there are datasets with significant and widespread violation of IIA. Hence, researchers interested in modeling choice data are well-served to perform tests for the presence of IIA.

(ii) We develop the first algorithm for nested logit tree induction, and show that its complexity is optimal. We show that the algorithm successfully recovers trees on synthetic data.

(iii) We evaluate our algorithm on a range of real datasets, and show both wins and losses, depending on how well nested logit actually captures user behavior in these domains. As a result, we suggest a set of open research directions for the future.

[2]We describe this two-stage process for model clarity; it is not generally believed that human decision-makers operate in this way.

2. PRELIMINARIES

We now review standard models from discrete choice theory that we will use throughout the paper. For a thorough treatment of the subject, see the books by Train [29] and Ben-Akiva and Lerman [3].

2.1 Multinomial logit

Under the multinomial logit model, the general choice theory framework stipulates making a selection from a set C of choices, where the utility of choice i is a random variable U_i:

$$U_i = V_i + \epsilon_i,$$

where V_i is the inherent *quality* and the ϵ_i are i.i.d. following a Gumbel distribution—formally, the density of the error is $f(\epsilon_i) = e^{-\epsilon_i - e^{-\epsilon_i}}$. It is not difficult to show that under a utility-maximization framework, the probability of selecting choice i is

$$P_i = \frac{e^{V_i}}{\sum_{j \in C} e^{V_j}}. \tag{1}$$

In other words, the probability of selecting an alternative is simply proportional to the exponential of the base utility of the item, e^{V_i}. This holds regardless of the choice set C.

While this formulation comes from discrete choice theory, multinomial logistic regression is also a widespread tool in machine learning [12]. In this domain, one usually has a small, fixed choice set C, a feature vector $x \in \mathbb{R}^p$, regression coefficients $\beta_i \in \mathbb{R}^p$ for each alternative in $i \in C$, and chooses item i with probability proportional to $e^{\beta_i^T x}$.[3]

2.2 Independence of irrelevant alternatives

A property of the multinomial logit model is the "independence of irrelevant alternatives" (IIA), i.e., the relative probability of someone choosing between two options is independent of any additional alternatives in the choice set. This is a straightforward observation from Equation 1:

$$\frac{P_i}{P_j} = \frac{e^{V_i} / \sum_{k \in C} e^{V_k}}{e^{V_j} / \sum_{k \in C} e^{V_k}} = \frac{e^{V_i}}{e^{V_j}}.$$

The IIA property dates back to Arrow's work in voting theory [1], and was first used in choice theory by Luce to derive Equation 1 [19].

Violations of IIA. There are a number of ways that the IIA assumption can be violated. Expanding on our example of liberals and conservatives from the introduction, we provide a thought experiment example following [20] based on our restaurant choice data. Consider someone who equally enjoys both traditional Japanese (TJ) and Italian (I) cuisine. Assume there is one restaurant of each type open for business. In this case, the user's preference is split evenly, so $P_{\text{TJ}}/P_{\text{I}} = 1$. Suddenly, a new sushi bar opens up and begins cannibalizing business from the Japanese restaurant. Our restaurant-goer still enjoys Italian food roughly half the time, but now chooses equally between the old Japanese restaurant restaurant and the new sushi bar. Her relative choice probability is now $P_{\text{TJ}}/P_{\text{I}} = 0.5$. IIA is violated in this setting because the relative probability changed with the introduction of the sushi alternative. Violations of the IIA assumption have been observed through explicit statistical tests on real-world preference data [15], preference experiments in psychology [24], and biological behavioral studies [18].

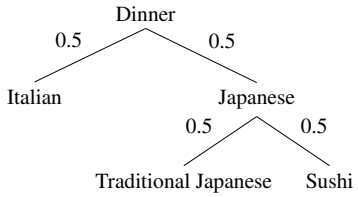
Dealing with IIA. While counter-examples to IIA as a choice axiom abound, we are more interested in:
(i) how often IIA is violated;
(ii) how to model the cases where IIA is violated; and
(iii) automatically constructing these models.

[3]It is standard practice to just consider the probabilities of the first $|C| - 1$ alternatives and then set the probability for the last choice so that the total probability sums to 1.

In classical discrete choice theory and machine learning with only a few alternatives and a fixed or lightly-varying choice set C, the first issue is of minor importance. In contrast, our datasets involve selections from a large number of alternatives and from many different choice sets C. Here, we can evaluate the frequency of IIA violations in terms of pairs over varying choice sets (see Section 3). To address the second issue, we turn to the nested logit model, described in the following section. Finally, we deal with computational issues in constructing the nested logit model, which arise from the large number of items that appear in choice sets (see Section 4).

2.3 Nested logit

A natural way to model the restaurant choice example in the prior section is to group the Japanese and sushi restaurants into a single category and use a sequential decision process captured by a tree:



The *nested logit model*, originally introduced by McFadden [21], encapsulates this process. In the nested logit, the choices are grouped into nests (clusters) such that IIA holds *within* a nest but not necessarily *between* nests. In our example, we have a Japanese nest consisting of the traditional Japanese and sushi restaurants. Conditioned on choosing between Japanese restaurants, IIA holds, but sushi and traditional Japanese *cannibalize* business from one another. We often refer to the traditional Japanese and sushi alternatives as *exchangeable*.

Formally, the nested logit model is given by a tree T with a designated root node. Each non-root node i has an associated utility V_i and edges are traversed from the root with probabilities relative to e^{V_i}.[4] Leaf nodes in the tree correspond to items that can appear in a choice set and internal nodes are nests of a set of items. If the choice set precludes an edge from traversal, the relative probability of that decision is set to 0.

Traditionally, the modeler must go through the following iterative process:

1. Specify a tree structure, using domain knowledge.
2. Learn the parameters of the model.
3. Test to see if the model is appropriate.

The number of trees grows exponentially in the number of items (leaf nodes). In large datasets, there are too many trees to consider, making this process infeasible. In this work, we solve this problem by showing how to automatically construct an appropriate model, i.e., a tree with edge traversal probabilities, essentially eliminating the need for input and supervision from the modeler.

3. HOW OFTEN IS IIA VIOLATED?

We now explore how often the independence of irrelevant alternatives holds in large discrete choice datasets from the Web (described in Section 3.2). To accomplish this, we will bring to bear standard methods in hypothesis testing.

[4]Technically, this model is called the *hierarchical logit* [32] and is slightly more general than the nested logit model derived from utility maximization. This is appropriate in our case because we want the most predictive tree while still modeling cannibalization. However, we will keep the nested logit terminology since it is more prevalent in the discrete choice literature.

3.1 Statistical tests for IIA violations

We first describe a battery of statistical tests that quantify IIA violations. All the tests are of the following flavor: under the null hypothesis that IIA holds, certain random variables should have the same distribution, which can be checked by statistical tests. The fact that our datasets comprise a variety of choice sets and alternatives make these tests applicable. Note that even though these tests are based on established principles in statistics, their application to checking for IIA violation is new. Before we derive our tests, we first set up some notation.

We assume our dataset consists of m item selections $s_1, \ldots, s_m$ from choice sets $C_1, \ldots, C_m$, where $s_k \in C_k$ and C_k is a subset of the set $\mathcal{I}$ of all possible items. For our datasets, it is often the case that $|C| \ll |\mathcal{I}|$, i.e., the choice sets contain just a few of the many possible items. Finally, let $I(\cdot)$ be the binary indicator function.

We will assume that each choice set appears several times, i.e., there are several i for which C_i are the same. Next, let $N_{i,j,C}$ be the number of times that item i or j is chosen from choice set C and let $X_{i,j,C} \le N_{i,j,C}$ be the number of times that i is chosen, conditioned on item i or j being selected. Formally,

$$N_{i,j,C} = \sum_{k=1}^{m} I(s_k \in \{i, j\},\ C_k = C), \quad X_{i,j,C} = \sum_{k=1}^{m} I(s_k = i,\ C_k = C).$$

Let $P_{i,j,C} = X_{i,j,C}/N_{i,j,C}$ be the probability of choosing i over j in choice set C. Denote by $\mathcal{A}_{ij}$ the collection of unique choice sets C for which item i or j is selected at least once, i.e.,

$$\mathcal{A}_{ij} = \{C_k \mid s_k \in \{i, j\}\}.$$

Finally, let P_{ij} be the overall probability of choosing i over j, i.e.,

$$P_{ij} = \frac{\sum_{C \in \mathcal{A}_{ij}} X_{i,j,C}}{\sum_{C \in \mathcal{A}_{ij}} N_{i,j,C}}.$$

All of our tests are based on the following simple observation:

OBSERVATION 3.1. *Under the null hypothesis that IIA holds for items i and j, $P_{i,j,C}$ is the same for any $C \in \mathcal{A}_{ij}$.*

We propose four different translations of this observation to statistical hypothesis tests.

Simultaneous binomial (SB). Our first test is based on interpreting $X_{i,j,C}$ as a sample from a binomial distribution $\text{Binom}(N_{i,j,C}, P_{i,j,C})$. Provided we have sufficient data,[5] we can use a χ^2 test on the null hypothesis that the $P_{i,j,C}$ is the same for all $C \in \mathcal{A}_{ij}$, directly using Observation 3.1:

$$\sum_{C \in \mathcal{A}_{ij}} \frac{\left(X_{i,j,C} - P_{ij} N_{i,j,C}\right)^2}{P_{ij} N_{i,j,C}} \sim \chi^2_{|\mathcal{A}_{ij}|-1}.$$

This test gives a way to measure the frequency of IIA violations. We can fix a significance level α and compute the fraction of pairs (i, j) for which the p-value of the χ^2 test statistic is less than α.

Multiple sample binomial (MSB). Based again on Observation 3.1, instead of simultaneously testing over all choice sets, we can instead compare a single choice set C to the rest. For items i and j and a choice set $C \in \mathcal{A}_{ij}$, let $N_{i,j,\overline{C}}$ and $X_{i,j,\overline{C}}$ be the occurrence counts in choice sets other than C:

$$N_{i,j,\overline{C}} = \sum_{k=1}^{m} I(s_k \in \{i, j\},\ C_k \neq C),\ X_{i,j,\overline{C}} = \sum_{k=1}^{m} I(s_k = i,\ C_k \neq C).$$

Let $P_{i,j,\overline{C}} = X_{i,j,\overline{C}}/N_{i,j,\overline{C}}$. Under the null hypothesis that IIA holds for items i and j, Observation 3.1 implies $P_{i,j,C} = P_{i,j,\overline{C}}$. Thus, we have samples from two distributions, $X_{i,j,C} \sim \text{Binom}(P_{i,j,C}, N_{i,j,C})$ and

[5]A common rule of thumb is that $X_{i,j,C}$ and $P_{ij}N_{i,j,C}$ are both at least 5. The χ^2 test may omit the choice sets C where this is not true but still include them in the estimate of P_{ij}. Furthermore, Fisher's exact test can be used when the sample sizes are small [11].

Table 1: Characteristics of the datasets. The choice sets in all datasets are of size 3, except for SFWORK, which has choice sets of size 3, 4, 5, and 6. The number of item pairs counts the number of (i, j) item pairs, where i and j co-occur in at least one choice set where i or j is selected.

Dataset	# selections	# items	# item pairs	# choice sets
RESTAURANTS	406K	3.77K	2.22K	25.6K
JAPANESECUISINE	42.5K	31	127	634
LASTFMARTISTS	146K	15.5K	3.31K	63.7K
LASTFMGENRE	96.5K	94	258	16.3K
SFWORK	5.00K	6	10	12

$X_{ij,\overline{C}} \sim \text{Binom}(P_{ij,\overline{C}}, N_{ij,\overline{C}})$, and we want to test if the success probabilities are the same. This lends itself to another χ^2 test statistic:

$$\frac{\left(X_{ij,C} - P_{ij}N_{ij,C}\right)^2}{P_{ij}N_{ij,C}} + \frac{\left(X_{ij,\overline{C}} - P_{ij}N_{ij,\overline{C}}\right)^2}{P_{ij}N_{ij,\overline{C}}} \sim \chi^2_1. \tag{2}$$

We now have a test for all pairs (i, j) and each choice set $C \in \mathcal{A}_{ij}$. This gives us the benefit of identifying which choice sets are causing IIA violations. In general, this is a significance test for 2×2 contingency tables, the nuances of which are covered in a survey by Yates [34].

Aggregated multiple sample binomial (AMSB). We can also aggregate the p-values for each choice $C \in \mathcal{A}_{ij}$ from the MSB test since Observation 3.1 says that the null hypothesis should hold for all C. To perform the aggregation, we use the Bonferroni correction [10]. Let $p_{ij,C}$ be the p-value from the test statistic in Equation 2. For a fixed significance level α, we reject the IIA null hypothesis for items i and j if

$$\min_{C \in \mathcal{A}_{ij}} p_{ij,C} \leq \alpha / |\mathcal{A}_{ij}|.$$

We argue that the Bonferroni correction is appropriate. Indeed, we are looking for clear evidence that any single choice set contains an alternative that will shift the probability of choosing i over j. This single strong effect is the exact scenario where the Bonferroni correction has the most power [6].

Choice set binomial (CSB). Lastly, we consider a test for IIA that considers two different choice sets $C, C' \in \mathcal{A}_{ij}$. Under the null hypothesis that IIA holds for items i and j, Observation 3.1 implies $P_{ij,C} = P_{ij,C'}$. Thus, we again have samples from two binomial distributions, $X_{ij,C} \sim \text{Binom}(P_{ij,C}, N_{ij,C})$ and $X_{ij,C'} \sim \text{Binom}(P_{ij,C'}, N_{ij,C'})$ and want to test if the success probabilities are the same. This again leads to a standard χ^2 test:

$$\frac{\left(X_{ij,C} - P_{ij}N_{ij,C}\right)^2}{P_{ij}N_{ij,C}} + \frac{\left(X_{ij,C'} - P_{ij}N_{ij,C'}\right)^2}{P_{ij}N_{ij,C'}} \sim \chi^2_1.$$

This test gives us a comparison over pairs of choice sets. We can measure the frequency of IIA violations over items i and j and pairs of choice sets in which they appear.

3.2 Data

We collected a variety of datasets for our experiments. The datasets are described below and their relevant characteristics are listed in Table 1.

RESTAURANTS. This dataset consists of clicks on restaurants from a panel displayed on the Google search page as a result of structured queries such as "dinner in Palo Alto." The panel lists three distinct restaurants (with links to a page about the restaurant), which we consider to be the choice set. To avoid position bias on the screen, we consider the items in the choice set as (restaurant, position) pairs. We say that a user makes a selection from the choice set if she clicks on one of the links to the page of the business entity. Furthermore, we only consider instances where the user clicks on a single link. The data is aggregated over instances, so we have no information about any particular user's tendency to select certain items. Finally, we only consider choice sets that appear at least 5 times and pairs of items that appear in at least 5 choice sets.

JAPANESECUISINE. This dataset is derived from the same restaurant data. Here, we consider the choice sets to be the cuisine type of the restaurant, and we restrict the data to choice sets consisting of three different Japanese cuisine types. (We also include more general cuisine types, such as "asian fusion" that may encompass Japanese food.) Again, clicks on a restaurant link are interpreted as a selection from the choice set. We only keep choice sets that appear at least 3 times and pairs of items that appear in at least 3 choice sets.

LASTFMARTISTS. This dataset comes from the listening habits of nearly 1000 users on the music streaming service last.fm [7].[6] On the service, users can select specific songs to listen to. Here, we say that a user makes a selection when they listen to three different songs from three different artists in a row and then plays one of the three songs on the next play. Thus, the choice sets are all of size three. We consider the choice at the level of the artist, so the selection is the artist. In order to eliminate any position bias, we consider the position in the sequence of the three songs as part of the item. We only keep choice sets that appear at least 3 times and pairs of items that appear in at least 3 choice sets.

LASTFMGENRE. This dataset comes from lifting the alternatives in the LASTFMARTISTS dataset to genre. We derive the genre through tags provided by the users.[7] We say that the genre of an artist is the most commonly provided tag by users, breaking ties by choosing the most frequently occurring tag overall in the dataset.

SFWORK. This is a travel mode choice dataset derived from transportation preferences of commuters going to work in San Francisco. The data was collected as part of a survey by the Metropolitan Transportation Commission of California and was analyzed by Koppelman and Bhat [15]. The items are transportation options: walking, biking, driving alone, taking public transit, and ride sharing. The choice sets for an individual are the transportation options available for her commute to work. This data is publicly available.

3.3 Empirical observations

We ran all of the statistical tests from Section 3.1 and measured the frequency of IIA violations at a significance level of $\alpha = 0.05$. (We omitted the SFWORK dataset, which has just a few choice sets.) Table 2 lists the frequency of violations at this significance level—we can expect 5% of the tests to be rejected under the null hypothesis. For the RESTAURANTS dataset, the rejection rate for the SB, MSB, and AMBSB is slightly above the 5% level, signifying that a small percentage of the data may be violating the IIA assumption. The JAPANESECUISINE and LASTFMGENRE datasets exhibit extreme levels of IIA violation—in both cases 30% of the SB tests are rejected. The LASTFMARTISTS dataset also exhibits IIA violations, although not quite at this extreme of a level.

Our largest dataset, RESTAURANTS, had the smallest number of IIA violations, hovering right about the 5% rejection rate. We explored a number of explanatory causes for the IIA violations using

[6]The data is available at http://dtic.upf.edu/~ocelma/MusicRecommendationDataset/lastfm-1K.html.
[7]The data is available at http://musicmachinery.com/2010/11/10/lastfm-artisttags2007/.

Table 2: Rejection rate of each hypothesis test at significance level 0.05 on our datasets. The JAPANESECUISINE and LASTFMGENRE datasets exhibit extreme violations of IIA.

Dataset	Test SB	MSB	AMSB	CSB
RESTAURANTS	0.087	0.066	0.076	0.041
JAPANESECUISINE	0.325	0.238	0.316	0.093
LASTFMARTISTS	0.106	0.102	0.129	0.049
LASTFMGENRE	0.300	0.143	0.284	0.094

Table 3: Rejection rate of each hypothesis test at significance level 0.05 on the filtered RESTAURANTS data, where choice sets that are susceptible to cannibalization (by analyzing metadata) are discarded. Interestingly, the rejection rates hardly change.

Filter	Fraction of data kept	Test SB	MSB	AMSB	CSB
None	1.0	0.087	0.066	0.076	0.041
Cuisine type	0.98	0.082	0.067	0.075	0.041
Price level	0.67	0.079	0.061	0.071	0.039
Star rating	0.94	0.083	0.063	0.070	0.040
Chain	0.83	0.087	0.064	0.073	0.041
Geography	0.94	0.087	0.066	0.072	0.041

additional metadata about the restaurants. Specifically, we applied various filters to the choice sets in order to remove possible scenarios where cannibalization in business might occur. In other words, we threw out choice sets C for items i and j if the third alternative k in C had an opportunity to cannibalize from i or j. Here, this means that k shares some feature with exactly one of i or j. We applied the following filters:

Cuisine type. Discard the choice set when i and j have different cuisine types and k has the same cuisine type as one of i or j. Restaurants are labeled with several of a set of 422 tags and we consider two restaurants to be of the same type if they share a tag.

Price level. Discard the choice set when i and j have different price levels (at least 1.0 on a scale of 1.0-5.0) and k is at a similar price level to i or j (difference less than 0.25).

Star rating. Discard the choice set when i and j have a different star rating (at least 1.0 on a scale 1.0-5.0) and k has a similar price level to i or j (difference less than 0.5).

Chain. Discard the cases when exactly one of i and j is a chain restaurant and k is also a chain restaurant.

Geography. Discard the choice set when k is much closer geographically to one of i or j. Specifically, let $\text{dist}(i, j)$ be the distance between i and j. Discard the choice set when $\text{dist}(i, j) > 3\text{km}$ and either $\text{dist}(i, k) < 0.25 \cdot \text{dist}(i, j)$ and $\text{dist}(j, k) > 0.75 \cdot \text{dist}(i, j)$ or $\text{dist}(j, k) < 0.25 \cdot \text{dist}(i, j)$ and $\text{dist}(i, k) > 0.75 \cdot \text{dist}(i, j)$.

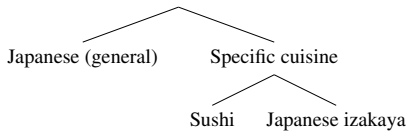
We note that we could also only run our tests on the choice sets where the cannibalization might occur. However, if the alternatives only cannibalize from either i or j, then this approach will not work. The rejection rate on each filtered dataset is provided in Table 3. Interestingly, the rejection rates hardly change for any of these filters. This suggests that obvious competition among restaurants such as type of food, quality, price, and geography are not actually creating violations of IIA. This can be interpreted as good news—a multinomial logit will be a good model for user behavior.

The statistical tests are also useful for identifying which pairs of items are most sensitive to alternative choice sets. The p-value exactly measures the likelihood of choices under the null hypothesis of IIA, so the smallest p-values correspond to the items that

Table 4: The 5 item pairs in the JAPANESECUISINE and LASTFMGENRE datasets that are the largest violators of IIA (least likely to satisfy IIA according to the p-value from the test statistic of SB). For the LASTFMGENRE dataset, the item format is genre_position, where position indicates the sequential slot in the three songs played.

Item pair	SB p-value
Japanese (general), sushi	1e-16
Okonomiyaki, teppan grill	1e-16
Asian, sushi	5e-12
Monja, Okonomiyaki	3e-11
Japanese (general), Japanese izakaya	1e-10
punk_1, rock_3	1e-16
punk_2, rock_3	1e-16
rock_1, electronic_3,	2e-10
rock_2, indie_3,	9e-09
rock_1, singer-songwriter_3	4e-07

are least likely to satisfy IIA. Table 4 lists the 5 item pairs with the smallest p-values for the SB test in the JAPANESECUISINE and LASTFMGENRE datasets. We can interpret these as the largest violators of IIA in the dataset. Interestingly, in the LASTFMGENRE dataset, "rock" appears in all of the top violators, whereas in the JAPANESECUISINE dataset, we see that "Japanese (general)" appears twice, once with sushi and once with Japanese izakaya. In this case, cannibalization due to a general category makes sense. The introduction of a third, descriptive alternative is likely to cannibalize from the existing descriptive cuisine type. This suggest the following tree:



First, the user decides on general Japanese food or a more specific cuisine type. Subsequently, users preferring more specific cuisine can choose from those alternatives. In the following sections, we discuss methods for automatically constructing these types of trees.

4. RECOVERING THE NESTED LOGIT TREE

We now switch gears and discuss how to recover nested logit structure in a dataset. In this section, we focus on theory and algorithms. In Section 5, we test the algorithms on synthetic datasets, and in Section 6 we test the algorithms on real-world datasets.

This section is organized as follows. We first define (Section 4.1) a powerful *tree oracle* which will reveal in a single (constant-time) query whether an item k *cannibalizes*[8] from either items i or j. Based on this oracle, we show (Section 4.2) how to recover a nested logit tree in quadratic time. We give a matching lower bound (Section 4.3), showing that this result is asymptotically optimal. We then show (Section 4.4) how the tree oracle may be implemented using queries, and observe that for nested logit trees that are well-separated in a technical sense defined below, the oracle requires only logarithmically many queries to the underlying choice process on slates of 2–3 items (Section 4.5). The quadratic lower bound continues to hold in this setting. We then give a simpler but slower

[8]We use the term cannibalization to encompass IIA violations, but there are a number of ways that violations can occur beyond those discussed in Section 3. See, for example, the work of Simonson and Tversky [26].

greedy algorithm (Section 4.6) that we implement in practice for our experiments. Finally, we show (Section 4.7) how to learn the edge weights of the tree to complete the specification of the nested logit process.

Before proceeding, we require some notation. Let $1, 2, \ldots, N$ denote the items in the dataset. We use the following convenient set representation of the nested logit tree. The leaf nodes are singleton sets consisting of a single item and internal nodes are sets whose elements are the children. With this set representation, the root node captures the entire tree structure. For instance, in our restaurant example from Section 2.2, the tree is represented by the root node $\{\{\text{Italian}\}, \{\{\text{Traditional Japanese}\}, \{\text{Sushi}\}\}\}$.

4.1 Tree oracle model

We now define the tree oracle used in our first algorithm. Section 4.4 discusses how to implement this oracle, and Section 4.5 shows how the oracle call may be implemented in logarithmic time for well-behaved trees.

Let $\text{lca}(i, j)$ denote the least common ancestor of nodes i and j in the tree and $\text{depth}(i)$ denote the depth of node i from the root. Given nodes i, j, and k, the tree oracle $\textsf{Oracle}(i, j, k)$ says which of the following is true:

1. `EQUAL`: k is an irrelevant alternative for i and j, which is true if and only if $\text{lca}(i, k) = \text{lca}(j, k)$.
2. `HIGHER`: k is a relevant alternative and cannibalizes from option i, which is true if and only if we have $\text{depth}(\text{lca}(i, k)) > \text{depth}(\text{lca}(j, k))$.
3. `LOWER`: k is a relevant alternative and cannibalizes from option j, which is true if and only if we have $\text{depth}(\text{lca}(i, k)) < \text{depth}(\text{lca}(j, k))$.

Here k can be either a single item (if it is a leaf node) or several items (if it is an internal node).

4.2 A quadratic-time algorithm

We now present a quadratic-time algorithm for recovering the nested logit tree assuming that the tree oracle is available. Later, we show in Proposition 4.1 that this algorithm is asymptotically optimal. Before we describe the algorithm, we introduce some additional notation. For any pair of leaf nodes i and j in the tree, define the sets of alternatives that cannibalize from either i or j:

$$S_i = \{\text{leaf nodes } k \mid \textsf{Oracle}(i, j, k) \text{ returns } \texttt{HIGHER}\},$$

$$S_j = \{\text{leaf nodes } k \mid \textsf{Oracle}(i, j, k) \text{ returns } \texttt{LOWER}\}.$$

We also define the set of sibling irrelevant alternative nodes whose least common ancestor with i and j is the same as $\text{lca}(i, j)$:

$$S_s = \{\text{leaf nodes } k \mid \textsf{Oracle}(i, j, k), \textsf{Oracle}(i, k, j) \text{ return } \texttt{EQUAL}\}.$$

Finally, we define the set of irrelevant alternative nodes for which the least common ancestor with nodes i and j is smaller in depth than $\text{lca}(i, j)$:

$$S_\emptyset = \{\text{leaf nodes } k \mid \textsf{Oracle}(i, j, k) \text{ returns } \texttt{EQUAL} \text{ and } \textsf{Oracle}(i, k, j) \text{ returns } \texttt{HIGHER}\}.$$

We note that for the leaf nodes $k \in S_\emptyset$, $\textsf{Oracle}(k, j, i)$ will return `LOWER`. These four subtrees provide a complete decomposition of the leaf nodes in the tree.

The basic steps of the reconstruction algorithm are as follows. First, choose any two leaf nodes i and j and identify the decomposition of leaf nodes into S_i, S_j, S_s, and $S_\emptyset$. Then, recursively construct the subtrees formed by S_i, S_j, and S_s. Finally, merge the subtree $S_i \cup S_j \cup S_s$ into a new leaf node. This merged node and the nodes in $S_\emptyset$ form a new set of leaf nodes, and the algorithm continues (see Figure 1). The above steps are constructed formally in Algorithm 1.

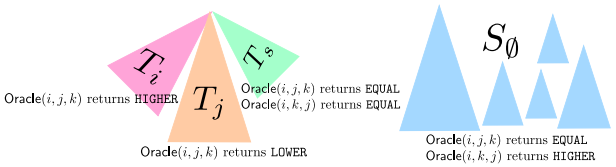


Figure 1: The decomposition of the leaf nodes (processed subtrees). Algorithm 1 recursively constructs the trees T_i, T_j, and T_s from the sets S_i, S_j, and S_s. These trees are merged into a new leaf node, which is included with the other leaf nodes in $S_\emptyset$.

Algorithm 1 Algorithm for recovering nested logit tree structure. The leaf nodes are items and each internal node is the set with elements as its children.

```
function ConstructTree(items 1, 2, ..., N)
    Leaf nodes T = {{1}, {2}, ..., {N}}.
    while |T| > 1 do
        Choose any two nodes i and j from T.
        S_i = {k ∈ T | Oracle(i, j, k) returns HIGHER}
        S_j = {k ∈ T | Oracle(i, j, k) returns LOWER}
        S_s = {k ∈ T | Oracle(i, j, k), Oracle(i, k, j) return EQUAL}
        S_∅ = {k ∈ T | Oracle(i, j, k) returns EQUAL,
                       Oracle(i, k, j) returns HIGHER}
        T_i = ConstructTree(S_i)
        T_j = ConstructTree(S_j)
        T_s = ConstructTree(S_s)
        T = {T_i, T_j, T_s} ∪ S_∅
    end while
    return T
end function
```

Complexity analysis. We now show that the algorithm only requires a quadratic number of queries to the oracle and a quadratic number of operations. The key components to the algorithm are (1) classify subtree nodes S_i, S_j, S_s, and $S_\emptyset$, (2) recurse on S_i, S_j, and S_s, (3) merge subtrees, (4) recurse on the remaining tree. As shown above, the first step requires a linear number of calls to the oracle. It takes at most linear time to form the node that contains the items in $S_i \cup S_j \cup S_s$, since it only needs to attach to at most a linear number of nodes (the children of the roots of S_i, S_j, and S_s). To analyze the recursion, let $T(N)$ be the time required to solve a problem instance with N leaf nodes and denote $N_i = |S_i|$, $N_j = |S_j|$, and $N_s = |S_s|$. The recursion satisfies

$$T(N) \le T(N_i) + T(N_j) + T(N_s) + T(N - N_i - N_j - N_s) + O(N).$$

Plugging in $T(N) = O(N^2)$ satisfies the inequality. We note that it is possible to improve the analysis in the average case, following the average-case analysis of, for example, quicksort [9].

4.3 A quadratic-time lower bound

We now show a lower bound that matches the algorithm: any deterministic tree reconstruction algorithm based on the oracle model must query the oracle at least a quadratic number of times.

PROPOSITION 4.1. *Any deterministic algorithm needs $\Omega(N^2)$ queries to the tree oracle to reconstruct the nested logit tree.*

PROOF. Suppose there exists an algorithm that uses $o(N^2)$ queries. Since there are N leaf nodes, there must be some pair of leaf nodes i and j such that there was no query that contains i and j as arguments. We claim that such an algorithm cannot distinguish between the following two trees:

(i) T_1 is a star: $\{\{1\}, \ldots, \{N\}\}$.
(ii) T_2 is the same as T_1 but replaces the edges from the root to i and j with an edge to $\{\{i\}, \{j\}\}$.
Indeed, any query containing node i as an argument with any two other arguments not equal to j returns EQUAL (the root is the lca). The same holds when j is an argument but not i. □

4.4 Implementing the tree oracle

To implement the oracle, we use classical hypothesis testing. Recall that the nested logit tree is defined by leaf nodes of singleton sets of the entire item population and internal nodes that correspond to the union of their children. We first consider queries that involve only leaf nodes in the tree (i.e., queries about individual items). We define P_{ij} to be the pairwise preference of item i over item j, i.e.,

$$P_{ij} = \Pr(s(C) = i \mid C = \{i, j\}),$$

where $s(C)$ is the random selection over the choice set C. The oracle implementation is based on the following simple observation.

OBSERVATION 4.2. *Alternative k is an irrelevant alternative of i and j if and only if including it in the choice set does not affect the pairwise preference of item i over item j. Formally,* Oracle(i, j, k) *returns* EQUAL *if and only if*

$$P_{ijk} := \Pr(s(C) = i \mid s(C) \in \{i, j\},\ i, j, k \in C) = P_{ij}. \quad (3)$$

We do not know whether Equation 3 holds, but we do observe samples $X_{ijk} \sim \text{Binom}(P_{ijk}, N_{ijk})$ and $X_{ij} \sim \text{Binom}(P_{ij}, N_{ij})$, where

$$X_{ijk} = \sum_{l=1}^{m} I(i, j, k \in C_l) \cdot I(s_l = i) \qquad N_{ijk} = \sum_{l=1}^{m} I(i, j, k \in C_l)$$

$$X_{ij} = \sum_{l=1}^{m} I(C_l = \{i, j\}) \cdot I(s_l = i) \qquad N_{ij} = \sum_{l=1}^{m} I(C_l = \{i, j\}).$$

With these observations, we can again rely on classical hypothesis tests to determine a confidence level for $P_{ij} = P_{ijk}$. In particular, we use the χ^2 test for the MSB test detailed in Section 3.1. Practically, we can fix some significance level α, and Oracle(i, j, k) returns EQUAL if the p-value is greater than α.[9] If the test rejects at the α level, the oracle returns HIGHER if $P_{ij} < P_{ijk}$ and LOWER if $P_{ij} > P_{ijk}$. Alternatively, we can also apply significance tests to the null hypotheses $H_{\text{HIGHER}}: P_{ij} < P_{ijk}$ and $H_{\text{LOWER}}: P_{ij} > P_{ijk}$.

When querying the oracle with sets of items, we have to be slightly more careful about the test. Let I, J, and K be sets of items with $i \in I$, $j \in J$, $k \in K$. We have to avoid contamination of the estimate P_{ijk} from alternatives that appear in I or J. The reason is that the presence of these alternatives may affect the probability of selecting item i over item j. We compare P_{ij} against the estimate for P_{ijk} that discards all choice sets C containing elements of I or J other than i and j. Formally, we estimate P_{ijk} by

$$P_{ijk} := \Pr(s(C) = i \mid s(C) \in \{i, j\},\ k \in C,\ C \cap I = \{i\},\ C \cap J = \{j\}).$$

Again, we observe a sample of a binomial with success probability equal to P_{ijk} and run the same χ^2 test.

4.5 Well-separated nested logit trees

We now observe that for "well-behaved" trees, a nearly-quadratic time algorithm can recover the nested logit tree using only the ability to ask for a random choice from a slate of 2–3 items.

Note that the goal of the oracle is to differentiate situations in which $P_{ijk} = P_{ij}$ from situations in which they are not equal. We say a nested logit tree is *ϵ-well-separated* if for all i, j, k we have $|P_{ijk} - P_{ij}| \notin (0, \epsilon]$. That is, the value of P_{ijk} is either identical to P_{ij} or different by more than ϵ. When ϵ is clear from context, we will simply refer to the tree as well-separated.

If a nested logit tree is well-separated, a constant number of choices $C(\epsilon, \delta)$ are sufficient to determine the tree oracle's response for i, j, k with failure probability at most δ. Hence, given $O(\log n)$ choices per oracle invocation, Algorithm 1 may be implemented with only $O(n^2 \log n)$ choices, with overall failure probability (inverse) polynomially small in n.

4.6 Greedy algorithm

We now present an alternative greedy algorithm for constructing the nested logit tree. We introduce this algorithm for the following reasons. First, and most importantly, it is more effective on sparse, real-world datasets where we do not have sufficient samples to completely implement the oracle (see Section 6.1). Second, it is considerably simpler to implement than Algorithm 1.

The greedy algorithm is based on the idea that it is simple to test whether or not two leaf nodes are siblings in the nested logit tree. This is described by the following proposition.

PROPOSITION 4.3. *Leaf nodes i and j are siblings in the nested logit tree if and only if* Oracle(i, j, k) *returns* EQUAL *for all other leaf nodes k.*

PROOF. If i and j are siblings and k is any leaf node, then we have $\text{lca}(i, k) = \text{lca}(\text{parent}(i), k) = \text{lca}(\text{parent}(j), k) = \text{lca}(j, k)$ and therefore Oracle(i, j, k) will return EQUAL. Now suppose that i and j are not siblings. Consider the siblings of node i. If i has all leaf siblings, then for any such one k, Oracle(i, j, k) will return HIGHER. Otherwise, i has internal nodes as siblings. Each of these subtrees has at least two leaf nodes. If j is in one of the subtrees, then for any leaf node $k \neq j$ in that subtree, Oracle(i, j, k) will return LOWER. If j is not in one of the subtrees, then for any leaf node k from any of the subtrees, Oracle(i, j, k) will return HIGHER. □

The greedy algorithm finds all siblings of some node i, merges these nodes into a new leaf node, and continues recursively. Algorithm 2 formally describes the procedure. We note that this algorithm can be implemented by an alternative oracle that implements the IsSibling function that determines whether or not two nodes in the tree are siblings. We will use this fact to implement the algorithm on sparse data. Even though the worst-case complexity of the greedy algorithm is $O(N^4)$, for the low-depth trees we encounter in our data, it is very efficient.

4.7 Learning edge probabilities

Once we have recovered the nested logit tree T, we must determine the probability of selecting an item given a set of alternatives. Recall that we can represent each edge in T as a relative probability. The choice set determines which edges in the tree can be traversed.

We represent each selection s_i from choice set C_i as a collection of d edge selections in T, $(s_{i1}, C_{i1}), \ldots, (s_{id}, C_{id})$, where $d = d(s_i)$ is the depth of the item in the tree. The probability of traversing each edge f is e^{V_f}, where V_f is the utility of the head node of the edge. Consequently, we learn a multinomial logistic model for each node in the tree, where the choice sets vary over the samples. The negative log-likelihood over the data is

$$-LL(\{(s_{ij}, C_{ij})\} \mid \{V_f\}) = -\sum_{i=1}^{m} \sum_{j=1}^{d(s_i)} V_{s_{ij}} + \log\left(\sum_{k \in C_{ij}} e^{V_k}\right). \quad (4)$$

Equation 4 is the sum of a linear term and a log-sum-exponential term on a subset of the variables, which is well-known to be convex over the domain $\mathbb{R}^N$ [5].

[9] If $\alpha = 0$, the null hypothesis is never rejected and Algorithms 1 and 2 recover a multinomial logistic model.

Algorithm 2 Greedy algorithm for recovering nested logit tree structure. The leaf nodes are items and each internal node is the set with elements as its children.

```
function GreedyConstructTree(items 1, 2, ..., N)
    Leaf nodes T = {1, 2, ..., N}.
    while |T| > 1 do
        Choose any node i from T.
        N = {j ∈ T | IsSibling(i, j)}
        if |N| > 1 then
            T = (T \ N) ∪ {N}
        end if
    end while
    return T
end function
function IsSibling(nodes i, j)
    for k ∈ T \ {i, j} do
        if Oracle(i, j, k) ≠ EQUAL then
            return false
        end if
    end for
    return true
end function
```

To learn the parameters V_f, we use gradient descent on the negative log-likelihood in Equation 4. Let $C(f)$ be the set of all choice sets in which edge f is an option, and let N_f be the number of times that f is traversed—formally, $C(f) = \{C_{ij} \mid f \in C_{ij}\}$ and $N_f = \sum_{C_{ij} \in C(f)} I(s_{ij} = f)$. Then the partial derivatives with respect to the log-likelihood are given by

$$\frac{\partial LL}{\partial V_f} = -N_f + e^{V_f} \sum_{C_{ij} \in C(f)} \frac{1}{\sum_{k \in C_{ij}} e^{V_k}}.$$

5. EXPERIMENTS ON SYNTHETIC DATA

We now test our recovery algorithms on synthetic data. Successful recovery requires sufficient fidelity to the data. This requirement is two-fold. First, we need pairwise and multi-way preference data. By pairwise preference data, we mean that for items i and j, we require knowledge about the probability of selecting each item if these are the only available options. Similarly, multi-way preference data are selections among more than two items. Second, there must be sufficient samples to use the hypothesis tests of Section 4.4. We employ Algorithm 2 to recover synthetically generated data and empirically evaluate the relationship between the number of samples and the degree of correctness in the recovered tree. In real-world datasets, we may not meet these requirements on the data in order to guarantee recovery. We defer to Section 6 for dealing with these issues.

Figure 2 shows the tree structure and edge probabilities of our synthetic tree. To generate a sample from the tree, we first uniformly at random choose the size of the choice set from $\{2, 3, 4\}$. After, we uniformly at random choose the items that appear in the choice set from the item set $\{A, B, C, D, E, F, G, H, I\}$. Finally, the selection is chosen according to the edge probabilities in the tree. We used this procedure to generate four datasets of different sizes. The number of selections were βN^2, where $N = 9$ is the number of items in the data set and $\beta = 10, 100, 500, 1000$.

We used the oracle with significance level $\alpha = 0.05$ on each synthetic dataset. The recovered trees are in Figure 3. We see the effect of samples size on the recovery of the tree. When $\beta = 10$, there is too little data to recover the tree. When $\beta = 100$, the tree recovers the nest structure of $\{A, B, C\}$, $\{D, E\}$, and $\{H, I\}$. However, there is too little data for selections of item G, and it is put in the nest with A, B, and C. The tree recovered when $\beta = 500$ is nearly the true tree, but is missing the extra hierarchy of the node $\{\{F, G\}, \{H, I\}\}$ that contains $\{F, G\}$ and $\{H, I\}$ as children. Finally, when $\beta = 1000$, we recover the full tree structure, up to small estimation errors in the edge probabilities.

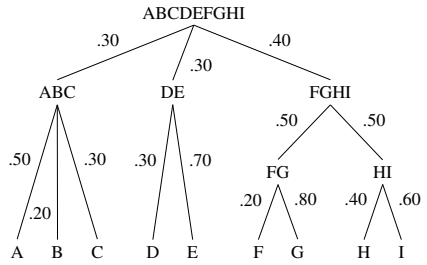


Figure 2: Tree for generating synthetic data. Figure 3 shows the trees recovered by our algorithm for various sample sizes.

6. EXPERIMENTS ON REAL-WORLD DATA

We now recover nested logit trees from real-world datasets. As discussed above, data sparsity changes the way in which we run the recovery algorithms, and we discuss these changes in Section 6.1. Our recovered trees reveal an interesting mixture of results.

6.1 Dealing with data sparsity

Compared to the synthetic data, our real-world datasets are sparse in several ways. First, we may lack the pairwise preferences to run the hypothesis tests described in Section 5. For example, the JapaneseCuisine dataset only contains selections from choice sets of three items. Second, we may not have sample coverage over all items. In other words, there may be items i, j, and k that do not co-occur in any choice set. Finally, we may not have sufficient samples to reject hypotheses at a meaningful significance level.

Nevertheless, we can still attempt to recover a tree using the greedy algorithm (Algorithm 2). The crucial insight is that we can still test whether or not two nodes are siblings in the tree, which is the only functionality needed by the greedy algorithm. The following observation is derived from Proposition 4.3:

Observation 6.1. *Two leaf nodes i and j are siblings in the nested logit tree if P_{ijk} is the same for all leaf nodes $k \neq i, j$.*

This observation is analogous to the derivation of the SB test in Section 3.1. Let K be the set of nodes k that appear in choice sets with i and j. Then we observe samples X_{ijk} from binomials $\text{Binom}(P_{ijk}, N_{ijk})$, where $P_{ijk} = \sum_{k \in K} X_{ijk} / \sum_{k \in K} N_{ijk}$, and we can employ the χ^2 test statistic used for the SB test.

This test tells us whether two leaf nodes i and j should be siblings, but it does not say whether or not the items corresponding to these nodes should be in their own nest or both be children of the root. We make one more observation that gives a necessary and sufficient condition for two leaf nodes belonging to a nest.

Observation 6.2. *In the nested logit model, sibling leaf nodes i and j are not children of the root if and only if there exists some node k such that* Oracle(i, k, j) *returns* `HIGHER` *or* Oracle(j, k, i) *returns* `HIGHER`.

Combining Observations 6.1 and 6.2 gives us a method to construct nests. To summarize, the process is:

IIA test: Check if the two items i and j satisfy IIA by looking at

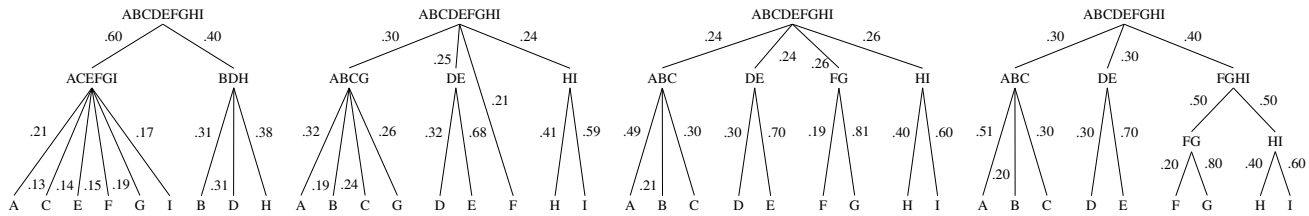


Figure 3: Recovered trees from synthetic data with βN^2 samples, $\beta = 10, 100, 500, 1000$ (left to right). As the number of samples increases, the recovered tree becomes closer to the underlying tree generating the data (see Figure 2).

Table 5: Negative log-likelihood for the recovered nested logit and multinomial logit on real-world datasets. The better model is in bold.

Dataset	Negative log-likelihood nested logit	multinomial logit
SFWORK	**4126**	4132
JAPANESECUISINE	**46217**	46272
LASTFMGENRE	67358	**62817**
SYNTHETIC (81K samples)	**82181**	85138

the probability of selecting i over j over all choice sets in which the items appear. This is formalized by *not rejecting* the null hypothesis that i and j satisfy IIA.

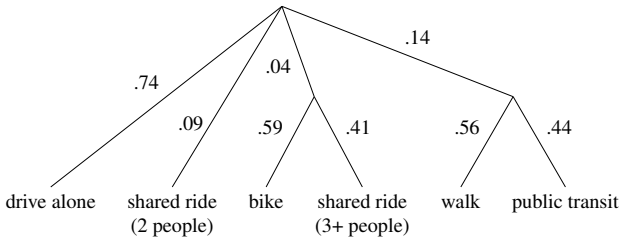
Cannibalization test: If IIA is not rejected, check if there is a third alternative k such that introducing j as an alternative decreases the probability of selecting i over k (or vice versa).

There are several practical heuristics that can be used with these tests. For example, we may require that the sibling nodes occur in at least q different choice sets. We can also set the significance level for the cannibalization test to be smaller than the significance level for the IIA test to be conservative about building nests.

6.2 Recovery results

Using the strategies for dealing with sparse data described above, we used the greedy algorithm to recover nested structure in our real-world datasets. Table 5 lists the negative log-likelihood of the recovered nested logit and multinomial logit for the SFWORK, JAPANESECUISINE, and LASTFMGENRE datasets. (We also list the corresponding results on the synthetic data for comparison.) We observe an interesting mix of results. Recovered nested logit has better likelihood for the SFWORK and JAPANESECUISINE dataset, but a worse likelihood for LASTFMGENRE dataset. Below, we discuss why this is the case and highlight some features of the recovered tree structures.

SFWORK. The recovered nested logit tree from the greedy algorithm is as follows.



The tree has two nests: one consisting of biking and shared rides between 3 or more people and the other consisting of walking and public transit. The latter nest can roughly be interpreted as "no equipment needed"—walking and public transit are the only options that do not require ownership of an item (car or bike). Interestingly, the two shared ride options do not appear in the same nest. This dataset suffers from data sparsity in the sense that the options available to individuals are correlated. For example, many people who can walk to work also have biking as an option.

JAPANESECUISINE. Figure 4 shows the recovered nested logit tree. The tree has natural nesting structure. For example, "Japanese (general)" and "regional Japanese" and "sushi" and "unagi" form nests, and these cuisine specifiers have natural exchangeability. The algorithm also discovers a nest consisting of "teppan grill" and "monja", two specific cuisines that are not obviously exchangeable. Overall, there is not much nesting structure—25 of the 31 cuisine types are just children of the root. Interestingly, natural nests such as "asian" and "pan asian" or "Japanese izakaya" and "modern izakaya" do not appear.

LASTFMGENRE. The greedy algorithm finds several levels of nesting structure, as illustrated in Figure 4. We see that several of the siblings are natural: classic rock and progressive rock, industrial and rock, singer-songwriter and alternative. We note that female vocalist and pop appear as siblings twice (for two different positions in the play sequence). Unfortunately, however, the likelihood of the recovered nested logit is still worse than the multinomial logistic. The reason is that these pairs are not completely exchangeable in the dataset. The nested logit tree forces us to assign a relative probability to a given nest. Although the siblings satisfy IIA and also exhibit cannibalization, we are really only guaranteed a likelihood improvement on choice sets that include *both* choices. Put another way, if i and j are siblings in the recovered nested logit tree, we will only see a likelihood improvement on the choice sets $C = \{i, j, k\}$ containing both i and j. In choice sets $C = \{i, k, l\}$ that only contain one of the siblings, we are forced to represent the probability of selecting i as the probability of selecting the nest $\{i, j\}$. Thus, the likelihood can be worse if the two items are not completely exchangeable.

In some sense, our algorithm is too greedy and forms nests when they may not be appropriate. One way to remedy this problem is to collapse any nest that does not improve likelihood. This would at least provide a guaranteed improvement over the multinomial logit model, which would arise from collapsing every nest. We leave efficient algorithms for this process as future work.

7. OTHER RELATED WORK

We now survey some additional related work. We first note that the nested logit is a popular model from a broader class of "generalized extreme value" models that also includes the paired combinatorial logit [31] and the heteroskedastic logit [28]. We choose to study the nested logit because of its simplicity, interpretability, and widespread use.

In the study of IIA, there are three classical statistical tests for the multinomial logit model. First, McFadden et al. [22] proposed

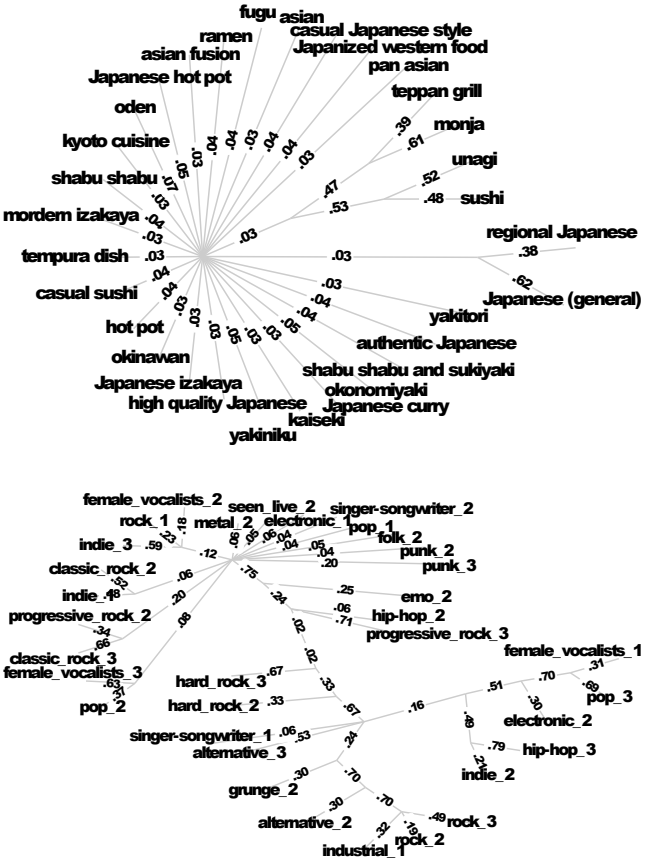


Figure 4: Top: Nest structure for the JAPANESECUISINE dataset. The tree contains sensible nests such as {sushi, unagi} and less obvious structure, such as the {teppan grill, monja} nest. The recovered nested logit model has better likelihood compared to the multinomial logit model (see Table 5). Bottom: Nest structure for the LASTFMGENRE dataset (leaf node children of the root are omitted). The names are of the format genre_position, where position indicates the position in the sequence of plays. The tree contains several sensible pairings (e.g., singer-songwriter_1 and alternative_3), but does not improve upon the likelihood of the multinomial logit model.

a likelihood ratio test on two models—a full model that accounts for all of the variables and a restricted model that only considers a subset. Small and Hsiao provide a modified version of the McFadden test to avoid asymptotic bias towards accepting the null hypothesis [27]. An alternative test proposed by Hausman and McFadden [13] uses a Hausman test [14] to compare the estimated parameters from the full and restricted models. Cheng and Long survey and empirically evaluate these tests [8].

These tests can be adapted to test whether a nested logit model satisfies the modified IIA assumptions governed by the nested logit tree [15]. However, these tests can only reject the validity of a fixed model. In other words, given a proposed nested logit tree structure, we can test whether or not the data follows that model. This gives no meaningful way to construct the nested logit tree, and a model must be proposed from intuition and then tested. In this work, we have presented an algorithm to automatically construct the nested logit tree from data. Other work in this direction includes learning Markov chain approximations for choice models [4].

There are also several studies of choice models for applications involving web data. Sheffet et al. predict when the click through rate of items on an e-commerce web site will change due to changes in the slate of alternatives displayed on the page [25]. Kumar et al. frame several user choice inference problems on the web as a problem of determining transition probabilities in a Markov chain when given the stationary distribution [17]. In the domain of recommender systems, Yang et al. employ user choice models to improve the performance of collaborative filtering algorithms [33]. Finally, through the analysis of map search logs, choice models have been developed to characterize how users select restaurants [16].

8. DISCUSSION AND CONCLUSIONS

In this paper we study the veracity of the IIA axiom in choice theory. We develop robust statistical tests for eliciting the presence of IIA and use it to analyze the prevalence of IIA in several real-world datasets. For the cases where IIA is violated, as in classical discrete choice theory, we resort to modeling the choice using a nested logit model. We develop an efficient algorithm to learn the tree representing the model by abstracting an oracle that answers queries about the structure of a tree. Under this oracle, we derived an optimal algorithm for the recovery of a nested logit tree. After recovering the tree structure, learning the remaining parameters of the model (the edge traversal probabilities) is a simple convex optimization problem. To our knowledge, this is the first efficient algorithm for learning a nested discrete choice model.

One crucial component to the robustness of our implementation of the oracle model was the availability of pairwise comparisons, i.e., user preferences when presented with a choice set of just two items. This is an important observation for experimental design when investigating discrete choice modeling. On the web, where experiments can be large and served automatically, this observation is particularly important.

The results of our algorithm on real-world datasets were mixed. In all cases, the recovered nests made sense in terms of item exchangeability; however, likelihood only improved in some of the datasets. The largest omission from our models is the variance in per-user preferences. Classical discrete choice models have addressed this problem theoretically, but the relevant issue is data sparsity—we do not have many samples for an individual. For example, the SFWORK dataset only contains one survey response per person and the JAPANESECUISINE dataset had no information about the users. One relevant direction for future work is to automatically cluster users and learn a nested logit model for each cluster. Alternatively, more sophisticated global models may also work well. We suspect that existing generalizations of the nested logit, such as overlapping nests [2, 30], could work well. However, it is not obvious how our formal results on nest recovery from the oracle model generalize to these models. This provides an interesting problem for future work.

Acknowledgments. We thank the reviewers for their suggestions.

9. REFERENCES

[1] K. J. Arrow. *Social Choice and Individual Values*. Yale University Press, first edition, 1951.

[2] M. Ben-Akiva and M. Bierlaire. Discrete choice methods and their applications to short term travel decisions. In *Handbook of Transportation Science*, pages 5–33. Springer, 1999.

[3] M. E. Ben-Akiva and S. R. Lerman. *Discrete Choice Analysis: Theory and Application to Travel Demand*. MIT Press, 1985.

[4] J. Blanchet, G. Gallego, and V. Goyal. A Markov chain approximation to choice modeling. In *EC*, pages 103–104, 2013.

[5] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[6] E. Candes. STATS 300C notes. http://statweb.stanford.edu/~candes/stats300c/Lectures/Lecture2.pdf. Accessed: Oct. 10, 2015.

[7] Ò. Celma Herrada. *Music Recommendation and Discovery in the Long Tail*. PhD thesis, Universitat Pompeu Fabra, 2009.

[8] S. Cheng and J. S. Long. Testing for IIA in the multinomial logit model. *Sociological Methods & Research*, 35(4):583–600, 2007.

[9] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, third edition, 2009.

[10] O. J. Dunn. Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293):52–64, 1961.

[11] R. A. Fisher. On the interpretation of χ^2 from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, pages 87–94, 1922.

[12] J. Friedman, T. Hastie, and R. Tibshirani. *The Elements of Statistical Learning*. Springer, 2001.

[13] J. Hausman and D. McFadden. Specification tests for the multinomial logit model. *Econometrica: Journal of the Econometric Society*, pages 1219–1240, 1984.

[14] J. A. Hausman. Specification tests in econometrics. *Econometrica: Journal of the Econometric Society*, pages 1251–1271, 1978.

[15] F. S. Koppelman and C. Bhat. A self instructing course in mode choice modeling: multinomial and nested logit models. *US Department of Transportation, Federal Transit Administration*, 31, 2006.

[16] R. Kumar, M. Mahdian, B. Pang, A. Tomkins, and S. Vassilvitskii. Driven by food: Modeling geographic choice. In *WSDM*, pages 213–222, 2015.

[17] R. Kumar, A. Tomkins, S. Vassilvitskii, and E. Vee. Inverting a steady-state. In *WSDM*, pages 359–368, 2015.

[18] T. Latty and M. Beekman. Irrational decision-making in an amoeboid organism: transitivity and context-dependent preferences. *Proceedings of the Royal Society of London B: Biological Sciences*, 278(1703):307–312, 2011.

[19] R. D. Luce. On the possible psychophysical laws. *Psychological Review*, 66(2):81, 1959.

[20] D. McFadden. Conditional logit analysis of qualitative choice behavior. In *Frontiers in Econometrics*. Academic Press, 1973.

[21] D. McFadden. Modelling the choice of residential location. In A. Karlqvist, L. Lundqvist, F. Snickers, and J. Weibull, editors, *Interaction Theory and Planning Models*, pages 75–96. North Holland, 1978.

[22] D. McFadden, W. B. Tye, and K. Train. An application of diagnostic tests for the independence from irrelevant alternatives property of the multinomial logit model. *Transportation Research Board Record*, pages 39–45, 1977.

[23] Nobel Media AB 2014. The Prize in Economic Sciences 2000 - Press Release. http://www.nobelprize.org/nobel_prizes/economic-sciences/laureates/2000/press.html. Accessed: January 31, 2016.

[24] C. Sedikides, D. Ariely, and N. Olsen. Contextual and procedural determinants of partner selection: Of asymmetric dominance and prominence. *Social Cognition*, 17(2):118–139, 1999.

[25] O. Sheffet, N. Mishra, and S. Ieong. Predicting consumer behavior in commerce search. In *ICML*, 2012.

[26] I. Simonson and A. Tversky. Choice in context: Tradeoff contrast and extremeness aversion. *JMR, Journal of Marketing Research*, 29(3):281, 1992.

[27] K. A. Small and C. Hsiao. Multinomial logit specification tests. *International economic review*, pages 619–627, 1985.

[28] J. H. Steckel and W. R. Vanhonacker. A heterogeneous conditional logit model of choice. *Journal of Business & Economic Statistics*, 6(3):391–398, 1988.

[29] K. E. Train. *Discrete Choice Methods with Simulation*. Cambridge University Press, 2009.

[30] P. Vovsha. The cross-nested logit model: Application to mode choice in the Tel-Aviv metropolitan area. *Transportation Research Board*, pages 6–5, 1997.

[31] C.-H. Wen and F. S. Koppelman. The generalized nested logit model. *Transportation Research Part B: Methodological*, 35(7):627–641, 2001.

[32] G. Y. Wong and W. M. Mason. The hierarchical logistic regression model for multilevel analysis. *Journal of the American Statistical Association*, 80(391):513–524, 1985.

[33] S.-H. Yang, B. Long, A. J. Smola, H. Zha, and Z. Zheng. Collaborative competitive filtering: learning recommender using context of user choice. In *SIGIR*, pages 295–304, 2011.

[34] F. Yates. Contingency tables involving small numbers and the χ^2 test. *Supplement to the Journal of the Royal Statistical Society*, pages 217–235, 1934.