LEARNING AN ARBITRARY MIXTURE OF TWO MULTINOMIAL LOGITS

WENPIN TANG

Abstract. In this paper, we consider mixtures of multinomial logistic models (MNL), which are known to $\epsilon$-approximate any random utility model. Despite its long history and broad use, rigorous results are only available for learning a uniform mixture of two MNLs. Continuing this line of research, we study the problem of learning an arbitrary mixture of two MNLs. We show that the identifiability of the mixture models may only fail on an algebraic variety of Lebesgue measure 0, implying that all existing algorithms apply in the almost sure sense. This is done by reducing the problem of learning a mixture of two MNLs to the problem of solving a system of univariate quartic equations. As a byproduct, we derive an algorithm to learn any mixture of two MNLs in linear time provided that a mixture of two MNLs over some finite universe is identifiable. Several numerical experiments and conjectures are also presented.

Key words: cubic equations, identifiability, mixture models, multinomial logits, multivariate polynomials, quartic equations, query complexity, symbolic computations.

1. Introduction

This paper is concerned with the problem of learning a mixture of two multinomial logistic models from data. Understanding an individual, or a user’s rational behavior when facing a list of alternatives is a classical topic in economic theory. In the era of data deluge, it has wide applications, especially for recommender systems where a user decides which of several competing products to purchase, and companies like Amazon, Netflix and Yelp look for which products are most relevant to a specific user. A powerful tool to study user behavior is discrete choice models, and we refer to McFadden [15, 16, 17, 18] for the modern literature. The most well-studied class of discrete choice models are the class of random utility models, which find roots in the work of Thurstone [25], and were formally introduced by Marschak [14]. The book of Train [26] contains a thorough review on this subject. Mixtures of multinomial logistic models (also known as mixed logits) are a family of parametric random utility models, which have been widely used since 1980 [2, 5], following earlier works of Bradley-Terry [3], and Luce-Plackett [13, 22] on the multinomial logits. Despite its broad use in practice, there are few works on efficient algorithms to learn any non-trivial mixture of multinomial logistic models. Chierichetti et al. [7] took the first step to develop polynomial-time algorithms to learn a uniform mixture of two multinomial logits. However, they pointed out that generalizing the results to non-uniform mixtures, or mixtures of more than two components is challenging.

The purpose of this paper is to go beyond the uniform mixture, and study the problem of reconstruction and polynomial-time algorithms to learn an arbitrary mixture of two multinomial logits. To proceed further, we give a little more background. A multinomial logistic
model, or simply an MNL over a universe \( \mathcal{U} \) is specified by a mapping from any non-empty subset \( S \subset \mathcal{U} \) to a distribution over \( S \). The set \( S \) is referred to as the choice set, from which a user selects exactly one item. The MNL requires a weight function \( w : \mathcal{U} \to \mathbb{R}_+ \) which gives a positive weight to each item in the universe \( \mathcal{U} \). The model then assigns probability to each \( u \in S \) propositional to its weight:

\[
P(u|S) := \frac{w(u)}{\sum_{v \in S} w(v)}, \quad \text{for each } u \in S.
\]

One can regard \( P(u|S) \) as the conditional probability of selecting item \( u \) given the alternatives in \( S \). It is often convenient to normalize the weight function \( w \) by \( \sum_{u \in \mathcal{U}} w(u) = 1 \), so \( w : \mathcal{U} \to \Delta_{|\mathcal{U}|-1} \) where \( \Delta_{|\mathcal{U}|-1} := \{ (a_1, \ldots, a_{|\mathcal{U}|}) \in \mathbb{R}^{|\mathcal{U}|}_+ : \sum_{i=1}^{|\mathcal{U}|} a_i = 1 \} \) is the \((|\mathcal{U}|-1)\)-simplex with \(|\mathcal{U}| \) the number of items in \( \mathcal{U} \). Given sufficient data of slates \( S \) with resulting choices of \( u \in S \), it is possible to estimate the weight \( w \) via maximum likelihood estimation. The underlying problem is convex, and is easy to solve by gradient methods.

In spite of simple interpretation and computational advantages, MNL is criticized for being too restrictive on the model behavior across related subsets, and thus lack of flexibility. This drawback is due to the fact that MNL is defined as a family of functions mapping any \( S \subset \mathcal{U} \) to a distribution over \( S \), based on a single fixed weight function. One way to resolve this issue is to remove the constraint that the likelihood of each item is always proportional to a fixed weight. The aforementioned random utility model does the job: it is defined by a distribution over vectors, where each vector assigns a value to each item of \( \mathcal{U} \). A user then draws a random vector from this distribution, and selects the item of \( S \) with largest value. McFadden and Train [19] observed that any random utility model can be approximated arbitrarily close by a mixture of MNLs. Thus, learning general random utility models reduces to learning mixtures of MNLs. This is the reason why mixtures of MNLs are widely recognized by practitioners. However, almost all existing learning approaches are empirical, and there are little provable results on learning non-trivial mixtures of MNLs. The only exception is [7], where the authors resolved positively the problem of learning a uniform mixture of MNLs. Here we take a further step to study a possibly non-uniform mixture of two MNLs, giving a few positive results on optimal learning algorithms.

The object of interest is specified by the triple \((a, b, \mu)\), where \( a, b : \mathcal{U} \to \Delta_{|\mathcal{U}|-1} \) are two weight functions, and \( \mu \in (0, 1) \) is the mixing weight. A \( \mu \)-mixture of two MNLs \((a, b)\) assigns to item \( u \) in the set \( S \subset \mathcal{U} \) the probability

\[
\mu \frac{a(u)}{\sum_{v \in S} a(v)} + (1 - \mu) \frac{b(u)}{\sum_{v \in S} b(v)}.
\]

The goal of the learning problem is to reconstruct the parameters \((a, b, \mu)\) from the oracle returning the distribution over items of the slate induced by the mixture. In this paper, we assume that the mixing weight \( \mu \in (0, 1) \) is known. So the problem consists of learning the weight functions \((a, b)\) in a \( \mu \)-mixture of MNLs. The main result of [7] showed that for \( \mu = 1/2 \), (i) if \(|\mathcal{U}| \geq 3\), any uniform mixture of two MNLs is identifiable in the sense that if the uniform mixtures of MNLs \((a, b)\) and \((a', b')\) agree on each \( S \subset \mathcal{U} \), then \( a = a' \) or \( a = b', b = a' \); (ii) there is an algorithm which learns any uniform mixture of two MNLs in \( O(|\mathcal{U}|) \) time. Theorem 2.1 below contains more detailed statements. The idea relies on the fact that any uniform mixture of two MNLs over a 3-item universe is identifiable, and one can reconstruct the weight functions by querying to 2- and 3-slates, i.e., subsets \( S \).
with \(|S| = 2, 3\). But this algorithm fails for a non-uniform mixture of two MNLs, since a non-uniform mixture of two MNLs on a 3-item universe is not necessarily identifiable (see Section 3 or [7, Theorem 3]). Nevertheless, the latter is ‘rarely’ the case. One main point of this paper is to delve into the following result.

**Theorem 1.1.** Let \(n := |U| \geq 3\), and \(\mu \in (0, 1)\). If the uniform mixtures of MNLs \((a, b)\) and \((a', b')\) agree on each \(S \subset U\), then

\[
(a, b) \neq (a', b') \implies R_n(a', b') = 0,
\]

where \(R_n\) is some polynomial specified later in the proof. Consequently, if \(a = (a_1, \ldots, a_n)\) and \(b = (b_1, \ldots, b_n)\) are drawn from two independent distributions with continuous density on the \((n - 1)\)-simplex \(\Delta_{n-1}\), then a \(\mu\)-mixture of MNLs \((a, b)\) is identifiable almost surely.

Theorem 1.1 shows that the identifiability of a mixture of two MNLs may only fail on some algebraic variety \(R_n = 0\). We refer to the books of Sturmfels [20, 24] for a gentle introduction to algebraic geometry with focus on the computational aspects. More important than the theorem itself is the way to construct the multivariate polynomials \(R_n\). As we will see later, this and the problem of learning a mixture of two MNLs essentially boils down to the problem of finding the common roots of some univariate quartic equations. Unlike [7], we adopt purely an equation-solving approach which is more natural and transparent. Besides the validity of the algorithms in [7] in the almost sure sense, we also devise an algorithm (Theorem 4.3) to learn a \(\mu\)-mixture of two MNLs in \(3|U|\) time provided that a \(\mu\)-mixture of two MNLs over some finite universe is identifiable. The algorithm takes half the time that the adaptive algorithm in [7] uses to learn a uniform mixture of two MNLs. The contributions of the paper are twofold:

- We show that the identifiability of the mixture model does not cause much a problem, and it may only fail on some ‘small’ algebraic variety of Lebesgue measure 0. This is important to develop further Baysian nonparametric models, e.g. both \(\mu\) and \((a, b)\) are random, and are drawn from some distributions. That is, we model user choice by a mixture of random MNLs.
- We show that learning a mixture of two MNLs is reduced to solving a system of univariate quartic equations. This gives a possible way to prove the identifiability of any mixture of two MNLs over some finite universe. Numerical experiments suggest that the latter be true on a 4-item universe, based on which Theorem 4.3 gives a linear-time learning algorithm.

The remaining issues, which seem to be technically challenging, are polynomial conditions in nature. We hope that this work will draw attention to experts of algebraic geometry and symbolic computations, so that advanced techniques in these domains can be used or developed to solve the conjectures in the paper.

To conclude the introduction, let us mention a few relevant references. There are a line of works discussing heuristic approaches to learn mixtures of MNLs by simulation [8, 10, 12, 26]. Mixtures of MNLs have also been studied in the context of revenue maximization by [1, 23]. More related to this work are [6, 21, 27, 28], where different oracles are assumed. We refer to [7, Section 2] for a more detailed explanation of the aforementioned references, and various pointers to other related works.
Organization of the paper. Section 2 provides background, and collects existing results related to mixtures of MNLs. Section 3 warms up with a discussion of learning a mixture of two MNLs over a 3-item universe. Section 4 studies the general problem of learning a mixture of two MNLs over a n-item universe. Section 5 gives the conclusion.

2. Preliminaries and existing results

This section provides background on the multinomial logit models, and recalls a few existing results. We follow closely the presentation in Chierichetti et al. [7]. Consider the n-item universe, whose items are labeled by \([n] := \{1, \ldots, n\}\). A slate is a non-empty subset of \([n]\), and a k-slate is a slate of size k.

A multinomial logit, or simply 1-MNL is determined by a weight function \(a : [n] \to \Delta_{n-1}\), where \(\Delta_{n-1} := \{(a_1, \ldots, a_n) \in \mathbb{R}^n_+ : \sum_{i=1}^n a_i = 1\}\) is the \((n-1)\)-simplex, with \(\mathbb{R}_+\) the set of nonnegative real numbers. In this choice model, given a slate \(T \subset [n]\), the probability that item \(i \in T\) is selected is given by

\[
D^a_T(i) = \frac{a_i}{\sum_{j \in T} a_j}, \quad i \in T. \tag{2.1}
\]

One can also take a weight function \(a : [n] \to \mathbb{R}_+\), and normalizing each \(a_i\) by \(\sum_{j=1}^n a_j\) will not affect the selection probability (2.1). A mixture of two multinomial logits, or simply 2-MNL \(A = (a, b, \lambda)\) is specified by two weight functions \(a, b\), and a mixing parameter \(\lambda > 0\) in such a way that the probability that item \(i \in T\) is selected in the slate \(T\) is

\[
D^A_T(i) = \frac{1}{1 + \lambda} \left( \frac{a_i}{\sum_{j \in T} a_j} + \frac{\lambda}{1 + \lambda} \frac{b_j}{\sum_{j \in T} b_j} \right), \quad i \in T. \tag{2.2}
\]

For later simplification, we use the parameter \(\lambda > 0\) instead of \(\mu := 1/(1 + \lambda) \in (0, 1)\) as the mixing weight. So given a slate \(T \subset [n]\), \(A\) first chooses the weight function \(a\) with probability \(1/(1 + \lambda)\) and \(b\) with probability \(\lambda/(1 + \lambda)\), and then behaves as the corresponding 1-MNL. For ease of presentation, we drop the superscript and write \(D_T\) instead of \(D^A_T\) if there is no ambiguity. If \(\lambda = 1\) or \(\mu = 1/2\), the choice model is called a uniform 2-MNL.

The problem is to reconstruct the parameters \(A = (a, b, \lambda)\) of the mixture model (2.2), assuming an oracle access to \(D_T(i)\) for all \(T \subset [n]\) and \(i \in T\). There are two main problems: (i) identifiability of the model parameters; (ii) computational complexity, i.e. the number of queries to the oracle \(D_T(\cdot)\) for reconstruction. [7] studied the uniform 2-MNL, and gave polynomial-time algorithms to reconstruct any uniform 2-MNL. Their results are summarized in the following theorem.

Theorem 2.1. [7] Let \(n \geq 3\), and \(A = (a, b, 1), A' = (a', b', 1)\) be two uniform 2-MNL over \([n]\). Then:

(i) \(A\) and \(A'\) agree on each \(T \subset [n]\), i.e. \(D^A_T = D^{A'}_T\) for each \(T \subset [n]\) if and only if \(a = a', b = b',\) or \(a = b', b = a'\).

(ii) Any adaptive algorithm for 2-MNL which queries to k-slates requires \(\Omega(n/k)\) queries, and any non-adaptive algorithm for 2-MNL which queries to k-slates requires \(\Omega(n^2/k^2)\) queries.

(iii) There is an adaptive algorithm to learn a uniform 2-MNL with \(6n + \mathcal{O}(1)\) queries to 2- and 3-slates, and there is a non-adaptive algorithm to learn a uniform 2-MNL with \(2n^2 + \mathcal{O}(n)\) queries to 2- and 3-slates.
3. Learning 2-MNL on the 3-item universe

In this section, we study a 2-MNL on the 3-item universe \{1, 2, 3\}. As mentioned in the introduction, assume that the mixing parameter \( \lambda > 0 \) is known. So we only need to reconstruct the 1-MNL weights \( a = (a_1, a_2, a_3) \) and \( b = (b_1, b_2, b_3) \). As pointed out in \cite{7}, for \( \lambda \neq 1 \) the oracle \( \{D_{\{1,2,3\}}(\cdot), D_{\{1,2\}}(\cdot), D_{\{1,3\}}(\cdot), D_{\{2,3\}}(\cdot)\} \) does not uniquely determine the weights \((a_1, a_2, a_3, b_1, b_2, b_3)\). The main point of Theorem 1.1 is that this situation rarely happens, and we can characterize the instances where the uniqueness fails. As will be seen in Section 4, the computation yielding the non-uniqueness characterization for \( n = 3 \) is a building block to study the identification problem of 2-MNL for \( n > 3 \). As an easy consequence of Theorem 1.1 and \cite[Theorems 5 \& 6]{7}, the following result shows that the query complexity lower bounds (see Theorem 2.1 (ii)) can be achieved for a random 2-MNL.

**Theorem 3.1.** Let \( n \geq 3 \), and consider a 2-MNL \( A = (a, b, \lambda) \) over \([n]\). Assume that \( \lambda > 0 \) is known, and \( a = (a_1, \ldots, a_n) \) and \( b = (b_1, \ldots, b_n) \) are drawn from two independent distributions with continuous density on the \((n-1)\)-simplex \( \Delta_{n-1} \). Then:

(i) There is an adaptive algorithm to learn a 2-MNL \( A \) with \( 6n + O(1) \) queries to 2- and 3-slates almost surely.

(ii) There is a non-adaptive algorithm to learn a 2-MNL \( A \) with \( 2n^2 + O(n) \) queries to 2- and 3-slates almost surely.

Now we deal with the non-uniqueness issue of a 2-MNL for \( n = 3 \). Let the oracle \( \{D_{\{1,2,3\}}, D_{\{1,2\}}, D_{\{1,3\}}, D_{\{2,3\}}\} \) be generated from the weights \((a', b') \in \Delta_2 \times \Delta_2\), e.g. \( D_{\{1,2\}}(1) = \frac{1}{1+\lambda} \frac{a_1'}{a_1'+a_2'} + \frac{\lambda}{1+\lambda} \frac{b_1'}{b_1'+b_2'} \). To simplify the notations, we denote \( C_T := (1+\lambda)D_T \). The problem involves solving the following system of equations:

\[
\begin{align*}
\frac{a_i}{a_i + a_j} + \frac{b_i}{b_i + b_j} &= C_{i,j}(\cdot) & \text{for } i, j \in [3] \text{ and } i \neq j, & \quad (3.1a) \\
\frac{a_i + \lambda b_i}{b_i + b_j} &= C_{\{1,2,3\}}(\cdot) & \text{for } i \in [3], & \quad (3.1b) \\
\frac{a_1 + a_2 + a_3}{b_1 + b_2 + b_3} &= 1. & \quad (3.1c)
\end{align*}
\]

So there are 6 unknowns and 11 equations, 6 from (3.1a), 3 from (3.1b) and 2 from (3.1c). Since \( \frac{a_i}{a_i + a_j} + \lambda \frac{b_i}{b_i + b_j} + \frac{a_i + a_j}{a_i + a_j + \lambda b_i + b_j} = 1 + \lambda \) and \( \sum_{i=1}^{3} (a_i + \lambda b_i) = 1 + \lambda \), there are 7 linearly independent equations. The following lemma provides a simple way to narrow down the possible solutions to (3.1).

**Lemma 3.2.** Assume that \((a_1, a_2, a_3, b_1, b_2, b_3)\) solves (3.1), and \( b_1 \neq \frac{C_{\{1,2,3\}}(1)}{1+\lambda} \). Then:

(i) Each of \( a_1, a_2, a_3, b_1, b_2, b_3 \) can be written as a simple function of \( b_1 \), which is specified by (3.2)–(3.3).

(ii) \( b_1 \) solves a quartic equation given by (3.4).

**Proof.** By (3.1b)–(3.1c), it is easy to see that \((b_1, b_2)\) determines the remaining variables by

\[
\begin{align*}
 a_1 &= C_{\{1,2,3\}}(1) - \lambda b_1, & a_2 &= C_{\{1,2,3\}}(2) - \lambda b_2, & b_3 &= 1 - b_1 - b_2, \\
 a_3 &= 1 - C_{\{1,2,3\}}(1) - C_{\{1,2,3\}}(2) + \lambda (b_1 + b_2). & (3.2)
\end{align*}
\]
Note that (3.1a) can be rewritten as
\[ \frac{a_i}{1 - a_k} + \lambda \frac{b_i}{1 - b_k} = C_{i,j}(i) \quad \text{for } i, j \in [3] \text{ and } i \neq j \]  
\[ (3.1a') \]

Specializing (3.1a) to \( i = 2, \ j = 3 \) and using (3.2) to express \( a_1, a_2 \) in terms of \( b_1, b_2 \), we get
\[ b_2 = \frac{(C_{(2,3)}(2) - C_{(1,2,3)}(1) + \lambda b_1 - C_{(1,2,3)}(2)) (1 - b_1)}{\lambda ((1 + \lambda) b_1 - C_{(1,2,3)}(1))} \quad : \quad D(b_1). \]  
\[ (3.3) \]

where \( N(b_1) \) is a quadratic function of \( b_1 \), and \( D(b_1) \) is linear in \( b_1 \). Further specializing (3.1a) to \( i = 1, \ j = 3 \) and using (3.2)–(3.3) to express \( a_1, a_2, b_2 \) in terms of \( b_1 \), we have
\[ C_{(1,2,3)}(1) ((1 - C_{(1,2,3)}(2)) D(b_1) + \lambda N(b_1)) (D(b_1) - N(b_1)) \]
\[ - (C_{(1,2,3)}(1) - \lambda b_1)(D(b_1) - N(b_1)) - \lambda b_1 ((1 - C_{(1,2,3)}(2)) D(b_1) + \lambda N(b_1)) = 0. \]  
\[ (3.4) \]

The first term on the l.h.s. of (3.4) is a quartic polynomial in \( b_1 \), and the other two terms are cubic polynomials in \( b_1 \).

The main point of Lemma (3.2) is to reduce the system of equations (3.1) to that only involving the 4-tuple \((a_1, a_2, b_1, b_2)\):
\[ \frac{a_1}{1 - a_2} + \lambda \frac{b_1}{1 - b_2} = C_{(1,3)}(1), \quad \frac{a_2}{1 - a_1} + \lambda \frac{b_2}{1 - b_1} = C_{(2,3)}(2), \]
\[ a_1 + \lambda b_1 = C_{(1,2,3)}(1), \quad a_2 + \lambda b_2 = C_{(1,2,3)}(2). \]  
\[ (3.5) \]

Moreover, solving the system of equations (3.5) is equivalent to solving a univariate quartic equation. We call the equations (4.2a)–(4.2b) the \((a_1, a_2, b_1, b_2)\)-system. Note that if \( b_1 = \frac{C_{(1,2,3)}(1)}{1 + \lambda} \) and \( b_2 \neq \frac{C_{(1,2,3)}(2)}{1 + \lambda} \), then similar to (3.3) we can express \( b_1 \) in terms of \( b_2 \), and hence all the other variables in terms of \( b_2 \) by (3.2). If \( b_1 = \frac{C_{(1,2,3)}(1)}{1 + \lambda} \) and \( b_2 = \frac{C_{(1,2,3)}(2)}{1 + \lambda} \), it is clear that all the other variables are uniquely determined by (3.2).

Recall that the values of \((C_{(1,2,3)}, C_{(1,2)}, C_{(1,3)}, C_{(2,3)})\) are generated from some weights \((a', b')\). This implies that the quartic equation \( P_{12}(b_1) = 0 \) given by (3.4) has a real root \( b'_1 \in [0, 1] \). Here the subscript ‘12’ indicates that the polynomial \( P_{12} \) is associated with the \((a_1, a_2, b_1, b_2)\)-system. Let
\[ Q_{12}(b_1) := \frac{P_{12}(b_1)}{(b_1 - b'_1)}, \]

which is a cubic polynomial whose coefficients are rational functions of \((a'_1, a'_2, b'_1, b'_2)\). Therefore, the identifiability of a 2-MNL on the 3-item universe, or equivalently the uniqueness of the solution to the system of equations (3.1) reduces to the problem (i) if the cubic polynomial \( Q \) has a real roots \( b''_1 \in [0, 1] \) and \( b''_1 \neq b'_1 \); (ii) if the corresponding 6-tuple \((a''_1, a''_2, a''_3, b''_1, b''_2, b''_3)\) given by (3.2)–(3.3) solves (3.1). Algorithmically, this is rather easy to verify: Cardano’s formula [4] solves any cubic equation. Then it suffices to check if the corresponding \((a''_1, a''_2, a''_3, b''_1, b''_2, b''_3) \in [0, 1]^6\), and if the equation \( \frac{a''_1}{a''_2 + a''_3} + \lambda \frac{b''_1}{b''_2 + b''_3} = C_{(2,3)}(2) \), which is part of (3.1) but not in (3.5) holds.

Now we show that it is rarely the case that the system of equations (3.1) has more than one solution \((a, b) \in \Delta_2 \times \Delta_2\). In fact, it is even true that (3.1) can barely have more than one solution \((a, b) \in \mathbb{R}^3 \times \mathbb{R}^3\).
Proof of Theorem 1.1 \((n = 3)\). Consider \(P_{12}(b_1), Q_{12}(b_1)\) associated with the \((a_1, a_2, b_1, b_2)\)-system, and \(P_{13}(b_1), Q_{13}(b_1)\) associated with the \((a_1, a_3, b_1, b_3)\)-system. Observe that the system \([3.1]\) has only one solution if the polynomials \(P_{12}\) and \(P_{13}\) have only one common root \(b_1 = \beta_1\), or equivalently the polynomials \(Q_{12}\) and \(Q_{13}\) do not have any common root. It is well known \([11, \text{Lemma 3.6}]\) that the latter holds if and only if

\[\text{Res}(Q_{12}, Q_{13}) \neq 0,\]

where \(\text{Res}(Q_{12}, Q_{13})\) is the resultant of \(Q_{12}\) and \(Q_{13}\), the determinant of a \(6 \times 6\) Sylvester matrix whose entries are the coefficients of \(Q_{12}\) and \(Q_{13}\). Recall that the coefficients of \(Q_{12}\), \(Q_{13}\) are rational functions of \((a', b') = (a_1', a_2', b_1', b_2')\). By letting \(R_3(a', b')\) be the polynomial corresponding to the numerator of \(\text{Res}(Q_{12}, Q_{13})\), we have

\[\{(a', b') \in \Delta_2 \times \Delta_2 : (3.1)\text{ has more than one solution}\} \subset \{(a', b') \in \mathbb{R}^3 \times \mathbb{R}^3 : R_3(a', b') = 0\}.\]

That is, the uniqueness of the solution to \([3.1]\) may fail only for those \((a', b')\) on the algebraic variety \(R_3(a', b') = 0\).

Of course, it is rather impossible to put down the expression of \(R_3\) by hand. With the help of \texttt{Mathematica}, we get an expression of \(R_3(a, b)/\lambda^6\) as follows:

\[
\begin{align*}
-(a^3a_2 - a_1^4a_2 + a_1^2 a_2^2 - 2a_1 a_2 - a_2^2 + \cdots) + a_2 b_1^3 b_2^2 \lambda^2 - 2a_1 a_2 b_1^3 b_2^2 \lambda^2 + a_1^2 a_2 b_1^3 b_2^2 \lambda^2 \\
(-a_3 + a_1 a_3 + a_3^3 + \cdots) + 2a_1 a_3 b_2^2 \lambda - a_1^2 a_3 b_2^2 \lambda^2) + \cdots + 48 \cdots \\
+(-a_2 + a_1 a_2 + \cdots) + \cdots - a_1^2 a_2 b_2^2 \lambda^3) + (\cdots)^3
\end{align*}
\]

But it seems that even \texttt{Mathematica} finds it challenging to expand, or to simplify the above large expression. We only know that the maximum degrees of \(a_1, a_2, a_3, b_1, b_2, b_3\) appearing in \(R_3\) are \(26, 9, 9, 23, 9, 9\) respectively.

To conclude this section, we go back to the cubic polynomial \(Q_{12}\). This polynomial has either 3 real roots, or 1 real root and 2 complex conjugate roots. Since we are only concerned with the real roots of \(Q_{12}\), it is natural to ask whether \(Q_{12}\) can have 3 real roots for some \(\lambda > 0\) and \((a', b') \in \Delta_2 \times \Delta_2\). The point is that if \(Q_{12}\) has only 1 real root, the analysis may further be simplified. It turns out that this question is subtle. It is known \([3]\) that \(Q_{12}\) has 3 real roots if the discriminant of \(Q_{12}\) is nonnegative. Note that the discriminant of \(Q_{12}\) is a function of \((a_1', a_2', b_1', b_2')\). For \(\lambda = 2\), the functions \texttt{FindInstance} and \texttt{NSolve} in \texttt{Mathematica} do not find any 4-tuple \((a_1', a_2', b_1', b_2') \in \Delta_1 \times \Delta_1\) such that the discriminant of \(Q_{12}\) is nonnegative. Moreover, the function \texttt{Maximize} finds numerically the maximum of the discriminant of \(Q_{12}\) over \(\Delta_1 \times \Delta_1\), which is \(-0.00317 < 0\). These observations suggest that \(Q_{12}\) have only 1 real root when \(\lambda = 2\). However, for \(\lambda = 5\) and \((a_1', a_2', b_1', b_2') = (0.0389099, 0.000870832, 0.0565171, 0.943483)\), \texttt{Mathematica} finds that \(Q_{12}\) has 3 real roots: 0.043916, 0.164599, 0.281671. Based on many experiments, we conjecture that there is a threshold \(\lambda_{\text{thres}} > 0\) such that for \(\lambda < \lambda_{\text{thres}}\) \(Q_{12}\) has only 1 real root whatever the values of \((a_1', a_2', b_1', b_2')\), and for \(\lambda \geq \lambda_{\text{thres}}\) \(Q_{12}\) can have either 1 or 3 real roots depending on the values of \((a_1', a_2', b_1', b_2')\).
4. Learning 2-MNL on the n-item universe

This section is devoted to the study of a 2-MNL over \([n]\) for \(n > 3\). The key idea is to reduce the system of equations over \(2n\) unknowns to \((a_i, a_j, b_i, b_j)\)-systems. This also gives a promising way to prove the identifiability of a 2-MNL on \([n]\) for some possible \(n > 3\). The following result records the general structure of learning a 2-MNL over \(n\) items. Recall the definitions of \(D_T\) and \(C_T\) for \(T \subseteq [n]\) from Section 3.

Proposition 4.1. Let \(n \geq 3\), and \(\mathcal{A} = (a, b, \lambda)\) be a 2-MNL over \([n]\). Assume that \(\lambda > 0\) is known. Then learning \(\mathcal{A}\) is equivalent to solving the following system of equations with \(2n\) unknowns \((a_1, \ldots, a_n, b_1, \ldots, b_n) \in \mathbb{R}_+^n\):

\[
\sum_{j \in T} a_i + \lambda \frac{b_i}{\sum_{j \in T} b_j} = C_T(i) \quad \text{for } T \subseteq [n] \text{ with } |T| \geq 2, \text{ and } i \in T, \quad (4.1a)
\]

\[
\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} b_i = 1. \quad (4.1b)
\]

So there are \(2 + 2^{n-1}n - n\) equations, and at most \(3 + 2^{n-1}(n-2)\) of these equations are linearly independent.

Proof. Note that for each \(T \subseteq [n]\) with \(|T| = k\), (4.1a) contributes \(k\) equations, and \(k-1\) of these equations are linearly independent. The result follows from the well known identities \(\sum_{k=1}^{n} k\binom{n}{k} = 2^{n-1}n\) and \(\sum_{k=1}^{n} (k-1)\binom{n}{k} = 1 + 2^{n-1}(n-2)\).

The following \((a_i, a_j, b_i, b_j)\)-system is an obvious extension of (3.5) to the \(n\)-item universe:

\[
a_i \left(\frac{1}{1 - a_j} + \lambda \frac{b_i}{1 - b_j}\right) = C_{[n] \setminus \{j\}}(i), \quad a_j \left(\frac{1}{1 - a_i} + \lambda \frac{b_j}{1 - b_i}\right) = C_{[n] \setminus \{i\}}(j), \quad (4.2a)
\]

\[
a_i + \lambda b_i = C_{[n]}(i), \quad a_j + \lambda b_j = C_{[n]}(j). \quad (4.2b)
\]

Similar to Lemma 3.2, solving the system of equations (4.1) boils down to solving a univariate quartic equation by using \((a_i, a_j, b_i, b_j)\)-systems.

Lemma 4.2. Assume that \((a_1, \ldots, a_n, b_1, \ldots, b_n)\) solves (4.1), and \(b_1 \neq \frac{C_{[1,2,3]}(1)}{1+\lambda}\). Then:

(i) If for each \(i\) there exists a set \(T(i)\) containing \(i\) such that \(b_i = \frac{C_T(i)}{1+\lambda}\), then \(a_i = C_{[n]}(i) - \frac{\lambda}{1+\lambda} C_T(i)\).

(ii) Otherwise, assume without loss of generality \(b_1 \neq \frac{C_T(1)}{1+\lambda}\) for any set \(T\) containing \(i\). Then each of \(a_1, \ldots, a_n, b_2, \ldots, b_n\) can be written as a simple function of \(b_1\), and \(b_1\) solves a quartic equation.

Proof. Part (i) is straightforward. For part (ii), consider the \((a_1, a_i, b_1, b_i)\)-system for all \(i \in \{2, \ldots, n\}\). By Lemma 3.2, \(a_1, a_i, b_i\) are fully determined by \(b_1\), and \(b_1\) solves the quartic equation \(P_{12} = 0\) associated with the \((a_1, a_2, b_1, b_2)\)-system. \(\square\)

Basically, Lemma 4.2 shows that the system (4.1), with approximately \(2^{n-1}n\) equations has at most 4 solutions. This simplification only makes use of \(2n\) equations, which involves those in (4.1a) with \(|T| = n - 1, n\). That is, queries to \((n-1)\)-slates and \(n\)-slates. Theorem 1.1 is then a consequence of Lemma 4.2.
Proof of Theorem 1.1 (general n). We follow the notations in Section 3 which proves the result for \( n = 3 \). Similarly, define the polynomials \( P_{ij}, Q_{ij} \) associated with the \((a_i, a_j, b_i, b_j)\)-system. Recall that the coefficients of \( P_{ij}, Q_{ij} \) are rational functions of \((a_i', a_j', b_i', b_j')\). By Lemma 4.2 if the system of equations (4.1) has more than one solution, then the polynomials \( Q_{12}, Q_{13}, \ldots, Q_{1n} \) have a common root. The latter implies that each pair \((Q_{1j}, Q_{1k})\) have a common root, which is equivalent to

\[
\text{Res}(Q_{1j}, Q_{1k}) = 0 \quad \text{for} \quad j, k \in \{2, \ldots, n\} \quad \text{and} \quad j < k,
\]

where \( \text{Res}(\cdot, \cdot) \) is the resultant of two polynomials. Let \( W_{1jk} \) be the multivariate polynomial in \((a_1', a_j', a_k', b'_i, b'_j, b'_k)\) corresponding to the numerator of \( \text{Res}(Q_{1j}, Q_{1k}) \). We have

\[
\{ (a', b') \in \Delta_{n-1} \times \Delta_{n-1} : (4.1) \text{ has more than one solution} \}
\]

\[
\subset \{ (a', b') \in \mathbb{R}^n \times \mathbb{R}^n : W_{1jk} = 0 \text{ for all } 2 \leq j < k \leq n \}.
\]

In particular, one can take \( R_4(a', b') = \sum_{2 \leq j < k \leq n} (W_{1jk})^2 \) so that (4.1) has the unique solution when \( R_4(a', b') \neq 0 \).

There are many ways to build \( R_n \) in Theorem 1.1. One can take \( R_n = \sum_{(j,k) \in S} (W_{1jk})^2 \) for \( S \) any subset of \( \{ (j, k) : 2 \leq j < k \leq n \} \), e.g. \( R_n = \sum_{2 \leq j < k \leq n} (W_{jk})^2 \) in the previous proof, or just \( R_n = W_{1jk} \) for some \( j, k \). Given \((a, b) \in \Delta_{n-1} \times \Delta_{n-1} \), it is relatively easy to check numerically if \( R_n(a, b) = 0 \). But as mentioned in Section 3 even in the simple case \( R_n = W_{1jk} \) (with 6 variables), \texttt{Mathematica} cannot output the expression of \( R_n \), let alone doing further symbolic computations.

Now we give a simpler \( R_n \) when \( n > 3 \). Consider the equations in (4.1a) relating only \((a_1, a_2, b_1, b_2)\). For \( n > 3 \), in addition to the \((a_1, a_2, b_1, b_2)\)-system there is one more:

\[
\frac{a_1}{a_1 + a_2} + \lambda \frac{b_1}{b_1 + b_2} = C_{\{1,2\}}(1). \quad (4.3)
\]

By injecting (3.2)–(3.3) into (4.3), we get \( \tilde{P}_{12}(b_1) = 0 \) with \( \tilde{P}_{12} \) another quartic polynomial. Let \( \tilde{Q}_{12}(b_1) = P_{12}(b_1)/(b_1 - b'_1) \) be the corresponding cubic polynomial. Another simple choice for \( R_n \) is the numerator of \( \text{Res}(Q_{12}, \tilde{Q}_{12}) \), which \texttt{Mathematica} outputs:

\[
-\lambda \left\{ -8 a_1^{15} a_2^{18} b_1^3 + 80 a_1^{16} a_2^{18} b_1^3 - 360 a_1^8 a_2^{18} b_1^3 - 1680 a_1^9 a_2^{18} b_1^3 + 2016 a_1^{10} a_2^{18} b_1^3 - 1680 a_1^6 a_2^{18} b_1^3 \\
+ a_2^{18} b_1^3 + 415 b_1^{15} a_2 + 22 a_1^{12} b_1^{15} b_2^{15} \lambda^8 - 121 a_1^{14} b_1^{15} b_2^{15} \lambda^8 + 109 a_1^{14} b_1^{15} b_2^{15} \lambda^8 - 43 a_1^{15} b_1^{15} b_2^{15} \lambda^8 + 10 a_1^{16} b_1^{15} b_2^{15} \lambda^8 \\
- a_1^{15} b_1^{15} b_2^{15} \lambda^8 - a_1^{17} b_1^{11} b_2^{15} \lambda^8 \right\}
\]

Such defined \( R_n \) is a polynomial in 4 variables, and is apparently simpler than the previous choices for \( R_n \). The expression of this \( R_n \) has more than 400,000 terms, and the maximum degrees of \( a_1, a_2, b_1, b_2 \) appearing in \( R_n \) are 25, 15, 25, 15 respectively. An interesting question is whether \( \{ R_n = 0 \} \cap (\Delta_1 \times \Delta_1) \) is empty for all \( \lambda > 0 \). If so, the system of equations (4.2)–(4.3) has a unique solution \((a_1', a_2', b_1', b_2')\) which implies that any 2-MNL over \([n]\) is identifiable for \( n > 3 \). Unfortunately, the following example shows it is not the case.
\[
A = 2; \quad x_1 = 2/5; \quad x_2 = 2/5; \quad y_1 = 3/10; \quad y_2 = 3/10;
\]
\[
\text{Solve } (a_1/(a_1 + a_2) + \lambda * b_1/(b_1 + b_2) = x_1/(x_1 + x_2) + \lambda * y_1/(y_1 + y_2) \&\& a_1/(1 - a_2) + \lambda * b_1/(1 - b_2) = x_2/(x_1 + x_2) + \lambda * y_1/(y_1 + y_2) \&\& a_1 + \lambda * b_1
\]
\[
= x_1 * y_1 + \lambda * y_1/(1 - y_2) \&\& a_2/(1 - a_1) + \lambda * b_2/(1 - b_1) = x_2/(1 - x_1) + \lambda * y_2/(1 - y_1) \&\& a_1 + \lambda * b_2
\]
\[
= x_1 + \lambda * y_1 \&\& a_2 + \lambda * b_2 = x_2 + \lambda * y_2, \quad \{a_1, a_2, b_1, b_2\}
\]
\[
\{(a_1 \rightarrow 5/19, a_2 \rightarrow 5/19, b_1 \rightarrow 7/19, b_2 \rightarrow 7/19), \quad \{a_1 \rightarrow 2/5, a_2 \rightarrow 2/5, b_1 \rightarrow 3/10, b_2 \rightarrow 3/18\}\}
\]

In principle, determining if a 2-MNL over \([n]\) is identifiable, or if the system of equations (4.1) has only one solution reduces to determining if a set of about \(2^{n-1}\) univariate cubic equations have a common root. The latter is equivalent to whether the resolvent of these cubic polynomials is zero, and this amounts to a fairly large number of multivariate polynomial equations on \((a', b')\), see e.g. [9] Section 7. So as \(n\) increases, the set of \((a', b')\) for which the system of equations (4.1) has more than 1 solution becomes more and more restrictive. It is believable that there is a threshold \(n_{\text{thre}} > 3\) such that (4.1) has only one solution for \(n \geq n_{\text{thre}}\). Based on many experiments, we conjecture that \(n_{\text{thre}} = 4\); that is, any 2-MNL over \([n]\) for \(n \geq 4\) is identifiable. We leave this puzzle to interested readers.

To finish, we show that if a 2-MNL over \([k]\) for some finite \(k\) is identifiable (and thus can be uniquely reconstructed), then we can learn a 2-MNL over \([n]\) for \(n \geq k\) with \(O(n)\) queries.

**Theorem 4.3.** Assume that \(\lambda > 0\) is known, and a 2-MNL \((a, b, \lambda)\) over \([k]\) for some finite \(k\) is identifiable. Consider a 2-MNL \(A = (a, b, \lambda)\) over \([n]\) with \(n \geq k\). Then there is an algorithm to learn \(A\) with \(3n + O(1)\) queries.

**Proof.** By hypothesis, \((a_1, \ldots, a_k, b_1, \ldots, b_k)\) must take the form:
\[
a_j = a_j^{[k]} \Sigma_a, \quad b_j = b_j^{[k]} \Sigma_b, \quad \text{for } j \in [k],
\]
where \((a_1^{[k]}, \ldots, a_k^{[k]}, b_1^{[k]}, \ldots, b_k^{[k]})\) is the unique solution to (4.1) with \(n = k\), and \(\Sigma_a, \Sigma_b \in [0, 1]\). Here \(\Sigma_a = \sum_{j=1}^{k} a_j\) and \(\Sigma_b = \sum_{j=1}^{k} b_j\) have yet to be determined. It requires \(O(1)\) queries to find \((a_1^{[k]}, \ldots, a_k^{[k]}, b_1^{[k]}, \ldots, b_k^{[k]})\). For instance, by Lemma 4.2, there are at most 4 possible solutions associated with the \((a_1, a_2, b_1, b_2)\)-system, and a quartic equation is easily solved by Ferrari’s method [4]. Then it suffices to check which one of these solutions satisfy other \(2^{k-1}(k-2) - 3\) equations. Next for each \(j \in [k+1, \ldots, n]\), consider the \((a_1, a_j, b_1, b_j)\)-system (4.2) which has 3 queries. By Lemma 4.2, \(b_j\) can be expressed in terms of \(b_1\). So the condition \(\sum_{j=1}^{n} b_j = 1\) determines \(\Sigma_b\). Similarly, by expressing \(a_j\) in terms of \(a_1\), the condition \(\sum_{j=1}^{n} a_j = 1\) specifies \(\Sigma_a\). In total, it requires \(O(1) + 3(n - k) = 3n + O(1)\) queries to learn \(A\). \(\square\)

In contrast to [7] (see Theorems 2.1) which learns a 2-MNL by querying 2- and 3-slates, Theorem 4.3 learns a 2-MNL by querying mostly \((n - 1)\)- and \(n\)-slates. Recall that the adaptive algorithm in [7] requires \(6n + O(1)\) queries to oracles \((D_{i,j}(\cdot), D_{i,j,k}(\cdot), D_{i,k}(\cdot), D_{i,j,k}(\cdot))\). The algorithm in Theorem 4.3 requires an even smaller number \(3n + O(1)\) of queries to oracle entries \(D_T(i)\).

5. Conclusion

In this paper, we study the problem of learning an arbitrary mixture of two multinomial logit. We have proved that the identifiability of the mixture models is less problematic,
since it may only fail on a set of parameters of Lebesgue measure 0. The proof is based on a reduction of the learning problem to a system of univariate quartic equations. As a consequence, we also proposed an algorithm to learn any mixture of two MNLs in linear time under the condition that a mixture of two MNLs over some finite universe is identifiable.

The paper also leaves a few problems for future research. For instance, (i) prove all the conjectures in Sections 3 and 4; (ii) consider the identifiability issue for both the mixing parameter \( \lambda \) and the weight functions \((a,b)\); (iii) study the problem of learning a mixture of more than two multinomial logits. We hope that our work will trigger further developments on the mixture models.

References

[1] J. Blanchet, G. Gallego, and V. Goyal. A Markov chain approximation to choice modeling. Oper. Res., 64(4):886–905, 2016.
[2] J. H. Boyd and R. E. Mellman. The effect of fuel economy standards on the US automotive market: an hedonic demand analysis. Transportation Research Part A: General, 14(5-6):367–378, 1980.
[3] R. Bradley and M. Terry. Rank analysis of incomplete block designs: I. The method of paired comparisons. Biometrika, 39(3/4):324–345, 1952.
[4] G. Cardano. Ars Magna or The Rules of Algebra. Dover Publications, Inc., New York, 1993. Translated from the Latin and edited by T. Richard Witmer.
[5] N. S. Cardell and F. C. Dunbar. Measuring the societal impacts of automobile downsizing. Transportation Research Part A: General, 14(5-6):423–434, 1980.
[6] F. Chierichetti, R. Kumar, and A. Tomkins. Discrete choice, permutations, and reconstruction. In Symposium on Discrete Algorithms, pages 576–586, 2018.
[7] F. Chierichetti, R. Kumar, and A. Tomkins. Learning a mixture of two multinomial logits. In International Conference on Machine Learning, pages 961–969, 2018.
[8] Y. Ge. Bayesian Inference with Mixtures of Logistic Regression: Functional Approximation, Statistical Consistency and Algorithmic Convergence. PhD thesis, Northwestern University, Evanston, USA, 2008.
[9] R. Goldman. Algebraic geometry and geometric modeling: insight and computation. In Algebraic Geometry and Geometric Modeling, Math. Vis., pages 1–22. Springer, Berlin, 2006.
[10] C. A. Guevara and M. E. Ben-Akiva. Sampling of alternatives in logit mixture models. Transportation Research Part B: Methodological, 58:185–198, 2013.
[11] J. Harris. Algebraic Geometry: A First Course, volume 133 of Graduate Texts in Mathematics. Springer-Verlag, New York, 1995.
[12] M. Hurn, A. Justel, and C. P. Robert. Estimating mixtures of regressions. J. Comput. Graph. Statist., 12(1):55–79, 2003.
[13] R. D. Luce. Individual Choice Behavior a Theoretical Analysis. John Wiley & Sons Inc., 1959.
[14] J. Marschak. Binary choice constraints on random utility indicators. In Stanford Symposium on Mathematical Methods in the Social Sciences, pages 312–329.
[15] D. McFadden. Conditional Logit Analysis of Qualitative Choice Behavior, pages 105–142. Academic Press, New York, 1974.
[16] D. McFadden. Econometric Models of Probabilistic Choice, pages 198–272. MIT Press, Cambridge, 1981.
[17] D. McFadden. Qualitative response models, pages 1–38. Econometric Society Monographs. Cambridge University Press, 1983.
[18] D. McFadden. Economic Choices. American Economic Review, 91(3):351–378, 2001.
[19] D. McFadden and K. Train. Mixed MNL models for discrete response. Journal of Applied Econometrics, 15(5):447–470, 2000.
[20] M. Michałek and B. Sturmfels. Invitation to Nonlinear Algebra. 2020+. Available at https://personal-homepages.mis.mpg.de/michalek/NonLinearAlgebra.pdf
[21] S. Oh and D. Shah. Learning mixed multinomial logit model from ordinal data. In Advances in Neural Information Processing Systems, pages 595–603, 2014.
[22] R. Plackett. The analysis of permutations. Applied Statistics, pages 193–202, 1975.
[23] P. Rusmevichientong, D. Shmoys, C. Tong, and H. Topaloglu. Assortment optimization under the multinomial logit model with random choice parameters. *Production and Operations Management*, 23(11):2023–2039, 2014.

[24] B. Sturmfels. *Solving systems of polynomial equations*, volume 97 of *CBMS Regional Conference Series in Mathematics*. Conference Board of the Mathematical Sciences, Washington, DC, 2002.

[25] L. Thurstone. A law of comparative judgment. *Psychological Review*, 34(4):273, 1927.

[26] K. E. Train. *Discrete choice methods with simulation*. Cambridge University Press, Cambridge, second edition, 2009.

[27] Z. Zhao, P. Piech, and L. Xia. Learning mixtures of Plackett-Luce models. In *International Conference on Machine Learning*, pages 2906–2914, 2016.

[28] Z. Zhao and L. Xia. Learning mixtures of Plackett-Luce models from structured partial orders. In *Advances in Neural Information Processing Systems*, pages 10143–10153, 2019.

Department of Industrial Engineering and Operations Research, UC Berkeley

E-mail address: wenpintang@stat.berkeley.edu