Plackett-Luce Regression Mixture Model for Heterogeneous Rankings

Maksim Tkachenko
School of Information Systems
Singapore Management University
maksim.tkatchenko@gmail.com

Hady W. Lauw
School of Information Systems
Singapore Management University
hadywlauw@smu.edu.sg

ABSTRACT

Learning to rank is an important problem in many scenarios, such as information retrieval, natural language processing, recommender systems, etc. The objective is to learn a function that ranks a number of instances based on their features. In the vast majority of the learning to rank literature, there is an implicit assumption that the population of ranking instances are homogeneous, and thus can be modeled by a single central ranking function. In this work, we are concerned with learning to rank for a heterogeneous population, which may consist of a number of sub-populations, each of which may rank objects differently. Because these sub-populations are not known in advance, and are effectively latent, the problem turns into simultaneously learning both a set of ranking functions, as well as the latent assignment of instances to functions. To address this problem in a joint manner, we develop a probabilistic graphical model called Plackett-Luce Regression Mixture or PLRM model, and describe its inference via Expectation-Maximization algorithm. Comprehensive experiments on publicly-available real-life datasets showcase the effectiveness of PLRM, as opposed to a pipelined approach of clustering followed by learning to rank, as well as approaches that assume a single ranking function for a heterogeneous population.

Keywords

Mixture model; Graphical model; Plackett-Luce; Heterogeneous Ranking; Learning to rank

1. INTRODUCTION

Learning to rank is a machine learning approach to rank objects based on their features [6]. It has found applications in many areas. In information retrieval [24], we would like to know which search result is more relevant to a query, and thus should be ranked higher. In recommender system [38, 27], it is important to determine which item is preferred by a user, and thus should be recommended to the user. Several natural language processing tasks may also involve ranking, such as text summarization [34] or keyphrase extraction [18].

The key idea behind learning to rank is to learn a ranking function that maps feature vectors to rank scores or rank orders. This function is learned from data consisting of rankings or ranked list of objects. An implicit assumption in many scenarios is that these rankings come from a homogeneous population. In other words, there is one way to rank objects based on their features, which is represented by a central ranking function. This assumption of a central ranking function may very well be applicable to some scenarios, such as homepage finding or named page finding, where most users would practically agree on the rankings.

Problem Yet there are other scenarios where there may be more than one way to rank objects based on their features. In this paper, we consider the problem of modeling rankings for a heterogeneous population. In such a population, there may be several sub-populations that rank objects differently. We call such sub-populations "preference groups". For instance, when shopping for cameras, consumers may have diverse preferences with respect to the attributes of a camera, and therefore varied ways for ranking cameras. Professionals may rank DSLRs highly for its customizability, while casual users may prefer point-and-shoot cameras for its portability. In a voting electorate [13, 14], there may be several preference groups that rank electoral candidates differently based on where they stand on issues. Thus a single ranking function would not be able to represent diverse preference groups in a heterogeneous population.

If only these preference groups were identifiable or known in advance, then the problem would devolve into employing learning to rank separately within each preference group independently. On the contrary, in many cases we merely observe the diverse rankings within a population. Discovering the preference groups is inherently part of the problem.

The problem can thus be informally stated as follows. Given a set of objects and their feature vectors, as well as a set of ranked lists defined over these objects, we seek to learn K latent preference groups and correspondingly K ranking functions, one for each preference group. The population of ranked lists is heterogeneous, i.e., there may be different permutations of the same set of objects in the data.

Approach One way to think about the problem is to consider it as an amalgamation of two requisite components: discovering the preference groups, and employing learning to rank within each group.

To discover the preference groups, it is not appropriate to employ clustering in the feature space. The reason is that

CIKM'16 , October 24 - 28, 2016, Indianapolis, IN, USA

ISBN 978-1-4503-4073-1/16/10...$15.00

DOI: http://dx.doi.org/10.1145/2983323.2983763

heterogeneity in our context concerns the variance in rankings over objects with similar features. Therefore, it is more relevant to consider clustering in the ranking space. For this, we turn to mixture models for ranking distributions.

While there are several models for estimating the distribution over rankings [29], as reviewed in Section 2, we build on the Plackett-Luce model [36, 25], which is widely applicable and lends itself to maximum likelihood estimation. It is based on Luce's Choice Axiom [26], which states that the probability of choosing one item over another is not affected by the presence or absence of other items in the pool. This axiom is frequently cited in economics for modeling consumer behavior when choosing one product over another [3]. Plackett-Luce model is characterized by a set of item-specific parameters, as described in Section 4. In this case, each preference group is associated with a set of Plackett-Luce parameters. The K preference groups could therefore be modeled as a mixture of K Plackett-Luce models [13, 14].

One limitation of a ranking model such as Plackett-Luce is that it is defined over a finite set of objects. Therefore, it does not generalize well to items not seen, or rarely seen, in the training data. This is where the learning to rank component comes in. Instead of learning item-specific parameters in each preference group (defined over items), we seek to learn a ranking function defined over features. There are at least two advantages to this modeling. For one, we would obtain better generalization from greater applicability to unseen items with similar features. For another, we would obtain better interpretability, as it may allow inspection of which features are important to each preference group.

While it is possible to think of the two components identified above as a pipeline, and we will explore this as well in the experiments, it is much more natural to consider them as two inherent components of a *unified joint model*. For one thing, the two components are mutually beneficial. Better clustering leads to better ranking functions due to more accurate reflection of preferences. Meanwhile, better ranking functions lead to better clustering, allowing better alignment of each ranked list to the closest preference group. Moreover, in a joint model, there is no need for two sets of parameters, one for clustering and another for learning to rank, and the parameters can be unified.

Contributions In this work, we make the following contributions.

- First, as far as we could ascertain, this is the first work to unify mixture modeling for ranking and learning to rank within a single framework, in the context of heterogeneous population of rankings (see Section 3).

- Second, we propose a joint model: Plackett-Luce Regression Mixture or PLRM model, described in Section 4. It is a probabilistic graphical model that discovers latent preference groups and their corresponding ranking functions. Furthermore, in Section 5, we describe its inference algorithm based on Expectation-Maximization.

- Third, in Section 6, through comprehensive experiments on several public datasets with varying heterogeneity, we show the effectiveness of the joint PLRM model vis-á-vis a pipeline model, as well as learning to rank models designed for homogeneous populations.

2. RELATED WORK

Here, we review related areas in the literature.

Probabilistic Models for Ranking This deals with learning probability distributions over permutations (i.e., rankings). The observations are rankings over items. It generally pays little, if any significant attention to features.

In this work, we build on the Plackett-Luce model, first introduced by Plackett [36] and Luce [25] independently. It expresses the probability of a permutation in terms of element-specific parameters. [17] describes a Bayesian approach for inferring its parameters. Beyond a single ranking model, subsequent works consider the notion of a mixture of Plackett-Luce models. For instance, [7] describes a nonparametric extension to model an infinite number of items, and clustered rankings via Dirichlet process mixtures. Others apply mixture models for profiling Irish electorates, including [13, 14]. In turn, [15] explores a mixture of Benter's models, which are generalized forms of Plackett-Luce by including dampening parameters. [43] addresses the question of identifiability of Plackett-Luce mixture, and proposes an efficient method to learn mixture of two Plackett-Luce models. These models are concerned with rankings alone, while our focus is on modeling ranking functions based on features.

Aside from Plackett-Luce, there are other paradigms for expressing distribution over rankings. Some are based on the notion of distances [10]. For instance, Mallows model [28] expresses the probability of a permutation in terms of its distance to a reference permutation. [2, 21, 32] consider a mixture of distance-based models. Yet another paradigm is Bradley-Terry [4, 11], based on pairwise comparisons.

Learning to Rank Learning to rank [9] deals with finding a function to rank elements based on their features. There are three broad categories. Pointwise learns a score for an element. Pairwise learns a binary classifier comparing two elements. Examples of pairwise approaches are SVM-Rank [19] and RankNet [5]. Listwise optimizes for a ranked list of elements, exemplified by Coordinate Ascent [33] and ListNet [6]. These learning to rank methods assume one central ranking function, while we model multiple latent ranking functions in the context of a heterogeneous population.

Rank Aggregation Rank aggregation [8] is concerned with aggregating multiple rankings into one consensus ranking. This comes up in applications such as meta-search [39] that combines the results from multiple search engines, or preference aggregation [40] that combines preferences of users. This is a different problem to ours, as its aggregation objective is different from our objective that seeks to resolve the observed rankings into a number of preference groups.

Others Another related work [16] relies on clustering instances in the feature space to obtain rankings. This is a distinct problem that clusters instances by similarity in features, rather than similarity in ranking functions. [1] proposes a regression model based on Plackett-Luce model, but their formulation aims to deal with categorical data. Their formulation is not intended to be used for ranking data, and is significantly different from our Plackett-Luce regression.

Collaborative filtering deals with deriving representations for users and items to estimate ratings [20]. Instead of ratings, some techniques are based on rankings [37, 41, 23]. Just as learning to rank has conventionally been recognized as a different problem from collaborative filtering, our work is also distinct in that we learn ranking functions based on features, rather than relying on user-specific parameters.

3. FORMULATION

We consider a set of M items of the same type. For instance, in the context of consumer choice, these may be products of a given category, e.g., digital cameras. For multimedia retrieval, these may be images to be ranked.

An item i is associated with a feature vector x_i in the D-dimensional space, $x_i \in \mathbb{R}^D$. For instance, cameras may have features such as sensor size, the presence of flash, weight and physical dimensions, etc. For images, the features may be gist descriptors and color histograms [35]. The collection of feature vectors of various entities is denoted $X = \{x_i\}_{i=1}^{M}$. For ease of reference, we list our notations in Table 1.

In addition to X, we are also given N ranked lists $R = \{r^{(n)}\}_{n=1}^{N}$, corresponding to N "judges". A judge n may rank a subset of items denoted $\bar{X}_n \subseteq X$. The corresponding ranking induced on $\bar{X}_n$ in the form of a permutation, without ties, is denoted $r^{(n)}$. When item i (with feature vector x_i) is placed in position j among items in $\bar{X}_n$, we have $r_i^{(n)} = j$. Position $j = 1$ is the highest, followed by position 2, etc. Equivalently, we write $r^{(n)}[j] = i$.

We further assume that these judges can be clustered into K preference groups. Each group is relatively homogeneous, whereby the ranking behaviors of judges within a group do not vary too much. In contrast, ranking behaviors across groups are heterogeneous. Two individual judges from different groups are likely to have different rankings $r^{(n)} \neq r^{(n')}$ over the same set of items $\bar{X}_n = \bar{X}_{n'}$. These preference groups are latent, and need to be discovered from the data.

Problem Statement Our problem can thus be stated as follows. Given the feature vectors X and the ranked lists R, as well as an integer K, we seek to identify:

- K latent preference groups among the N judges in R,
- a ranking function within each latent preference group.

4. MODEL

In this section, we discuss the formal definition of the proposed Plackett-Luce Regression Mixture (PLRM) model. The plate representation of PLRM is shown in Figure 1.

Overview PLRM is a probabilistic graphical model for representing different latent preference groups within a population of judges. Each judge arranges a given set of items into a ranked list (a permutation) based on the features of the item. In the conventional Plackett-Luce model, the ranking is based on item-specific parameters, which may connote for item quality. In contrast PLRM assumes that the ranking is based on a ranking function on item features.

We further assume that K groups exist within the population, and each group is associated with a ranking function. Each judge's ranking is based on the ranking function of the group it belongs to, while still allowing for some variance among group members. Accounting for this variance is best done through probabilistic modeling.

To generate the observed ranked lists R, we consider N experiments as follows. At each random trial, we ask a new judge to select a group. The group is chosen stochastically with a categorical variable $z_n \in \{1, 2, ..., K\}$ indicating the choice. We then ask the judge to rank a subset of items, defined by their feature vectors $\bar{X}_n$. The judge relies on the group's ranking parameter $w_{z_n} \in \mathbb{R}^d$. This parameter is a vector in D-dimensional space, so that each component of w_{z_n} corresponds to a particular feature of $x_i \in \bar{X}_n$.

Table 1: Notations

Notation	Description
i	index of an item
M	total number of items
d	index of a feature
D	number of item features
x_i	feature vector of an item i
X	collection of items/feature vectors
n	index of a judge or a ranked list
N	total number of judges
$\bar{X}_n$	subset of items ranked by judge n
$r^{(n)}$	permutation over $\bar{X}_n$ given by judge n
r_i	position of item i in the ranked list r
$r[j]$	index of the item occupying position j in r
R	collection of ranked lists by N judges
K	number of preference groups
k	index of a preference group
w_k	preference vector for group k
W	collection of preference vectors
v_i	ranking parameter for item i, equivalent to $\exp\left(x_i w^T\right)$ for PLRM
V	collection of ranking parameters
π	mixture proportion among preference groups
z_n	group assignment for judge n
Z	collection of group assignments

To produce the ranking $r^{(n)}$ over items in $\bar{X}_n$, the judge may apply the group parameter w_{z_k} via regression over the items in $\bar{X}_n$, i.e., $Y = \bar{X}_n {w_{z_n}}^T$. Relying on exact regression values may be unrealistic, given the likely variance among group members. Therefore, to account for the trial uncertainties and ranking deviation among the group members, the regression values serve as conditional parameters to a ranking probability model, as described in the following.

Ranking Probability Model We first describe a ranking model based on the basic Plackett-Luce, after which we introduce the regression-based Plackett-Luce in PLRM.

Let r be a ranking of M items, i.e., permutation of M indices. Plackett-Luce (PL) model defines a probability distribution over all possible rankings of M items. It is expressed in terms of item-specific parameters $V = \{v_i\}_{i=1}^{M}$; $v_i \geq 0$.

$$\mathrm{PL}(r|V) = \prod_{j=1}^{M} p_j(r|V), \tag{1}$$

where

$$p_j(r|V) = \frac{v_{r[j]}}{v_{r[j]} + v_{r[j+1]} + \cdots + v_{r[M]}} = \frac{v_{r[j]}}{\sum_{l=j}^{M} v_{r[l]}}. \tag{2}$$

The probability distribution yields an intuitive interpretation in the form of a ranking procedure. A judge generates a ranked list from the first position to the last position. $p_1(r|V)$ indicates the probability of placing an item $r[1] = i$, parameterized by v_i, in the first place. Having selected the item to occupy the first position, we repeat this procedure with the subsequent positions. $p_2(r|V)$ is the probability of placing another element $r[2] = i'$, parameterized by $v_{i'}$ in the second place, and so on. This procedure continues for all the items within a ranked list r. The joint probability of this process for a ranked list r is presented in Eq. 1.

The PL model defined above has a couple of important properties. The first one is the intuitive property that an item i is more likely to be placed higher than another item i', if $v_i > v_{i'}$. The second property flows from the aforementioned Luce's Choice Axiom. Items that have already been placed into r would not influence the choice probability of the remaining items. This property, also known as "independence from irrelevant alternatives" [26], allows ranked lists of varying sizes to be induced for subsets of items.

One limitation of the conventional PL model, in the context of learning to rank, is the reliance on the item-specific parameter v_i. This requires all items not just to have been seen, but also to have had sufficient representation in the training data. To address this limitation, we therefore seek to bring the ranking parameter into the feature space of items. This is accomplished by expressing the parameter v_i in terms of a regression of the feature vector x_i with weight parameter or "preference vector" w, as expressed in Eq. 3. In this work, we use the exponential transformation to satisfy non-negativity constraint. In practice, there could be other possible choices such as sigmoid.

$$v_i = \exp\left(x_i {w}^T\right) \tag{3}$$

We call this approach *Plackett-Luce Regression* or PLR. However, because of the heterogeneity of the population, there may not be only a single preference vector w for all ranked lists. Instead, we postulate that there are K subpopulations, or preference groups, each of which is associated with its own preference vector. This gives rise to a mixture of PLR models, which we term the *Plackett-Luce Regression Mixture* or PLRM, as described in the following.

Generative Process The PLRM model can effectively be described by the following generative process.

1. π, a K-dimensional mixture proportion, is sampled from Dirichlet distribution with symmetric prior α:
$$\pi \sim Dirichlet(\alpha)$$
2. For each of the K preference groups, its preference vector w_k is sampled from a D-dimensional Gaussian with zero mean and σ^2 variance:
$$w_k \sim \mathcal{N}(0, \sigma^2)$$
3. For each ranked list $r^{(n)}$ defined over the subset of items $\bar{X}_n \subset X$, where $n = 1, \ldots, N$:
   (a) Select a preference group z_n from a choice of K groups according to the mixture proportion π:
   $$z_n \sim Categorical(\pi)$$
   (b) Sample a ranking $r^{(n)}$ from the Plackett-Luce model parameterized by the regressed values over the set of feature vectors in $\bar{X}_n$:
   $$r^{(n)} \sim \mathrm{PL}\left(\exp\left(\bar{X}_n {w_{z_k}}^T\right)\right)$$

The likelihood of this generative process is as follows:

$$\mathcal{L}(R, Z, W, \pi | X) = \mathrm{P}(\pi|\alpha) \times \prod_{k=1}^{K} \mathrm{P}(w_k|0, \sigma^2) \times \prod_{n=1}^{N} \mathrm{P}(z_n|\pi) \mathrm{PL}\left(r^{(n)} | \exp\left(\bar{X}_n {w_{z_n}}^T\right)\right), \tag{4}$$

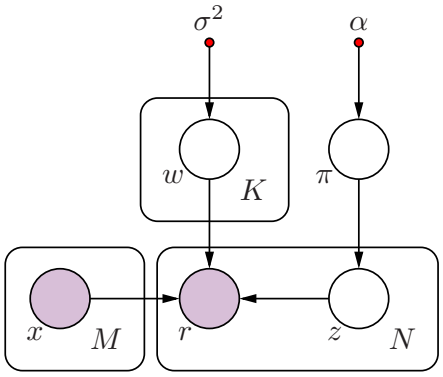


Figure 1: Plackett-Luce Regression Mixture Model in Plate Notation

where $R = \{r^{(n)}\}_{n=1}^{N}$ are the set of ranked lists, $Z = \{z_i\}_{n=1}^{N}$ are the corresponding group assignments for each ranked list, and $W = \{w_k\}_{k=1}^{K}$ are the groups' preference vectors.

Discussion The above generative process defines the probabilistic generative model that we call PLRM, with a mixture modeling component representing the latent preference groups as well as a regression component representing the learning to rank based on features. This represents the *joint* modeling approach. With appropriate settings, we can decouple the two components, yielding simpler models.

First, we can turn the model into a purely clustering model based on rankings, without features. In this case, an item i is represented by a feature vector x_i, whose dimensionality is the same as the number of items M. Rather than representing features, x_i becomes a one-hot "identity" vector, with a value of 1 in the i-th dimension, and 0 in all other dimensions. Effectively, the regression $x_i w^T$ yields an item-specific ranking parameter, just as in the original PL model. Given that this results in a mixture of K Plackett-Luce models, we call this *Plackett-Luce Mixture* or PLM, which is capable of clustering but not ranking by features. Later, we will compare PLRM to PLM, to verify that regression on the features does help the clustering function.

Second, we can turn the model into a purely learning to rank model, by simply setting $K = 1$. In this case, there is no mixture. There is only a single regression model, embedded within a probabilistic ranking model. We thus call this *Plackett-Luce Regression* or PLR, which is capable of learning to rank, but not clustering. Later, we will compare PLRM to PLR, to verify that modeling a mixture does help for a heterogeneous ranking population.

Third, the above two simpler models essentially decouple the two components that are joined together by PLRM. Therefore, they could be employed in a disjoint pipeline. This pipeline of PLM+PLR would first cluster the ranked lists in the population R into K preference groups using PLM, without the help of features. Thereafter, we run PLR within each preference group to learn a ranking function based on features. Later, we will compare PLRM to PLM+PLR to see how the joint approach compares in the effectiveness of both the clustering and ranking objectives.

5. INFERENCE

In this section, we derive an Expectation-Maximization (EM) algorithm for fitting the Plackett-Luce Regression Mixture (PLRM) model parameters, as well as discuss how the model could be used for ranking prediction.

5.1 Optimization

EM is an iterative algorithm that is commonly used for finding maximum likelihood estimate of a model involving unobserved parameters. In the case of PLRM, we consider the group assignments $Z = \{z_n\}_{n=1}^{N}$ as latent variables that guide the estimation procedure. The groups' preference vectors $W = \{w_k\}_{k=1}^{K}$, as well as the mixture proportion π, are unknown parameters to be maximized during the maximization step. The initial estimates are chosen randomly.

Expectation Step In the expectation step, we estimate the latent variables (Z), and calculate the expected value of the log likelihood function with respect to their a posteriori distribution. We denote the expected value of the log likelihood as follows:

$$Q(W, \pi | W', \pi') = E_{Z|R,W',\pi',X}\left[\log \mathcal{L}(R, Z, W, \pi | X)\right], \quad (5)$$

where W' and π' are the current parameter estimates.

Let T_{nk} be an auxiliary function defined as follows:

$$T_{nk} = \frac{\mathrm{P}(z_n = k|\pi')\mathrm{PL}\left(r^{(n)}|\exp\left(\bar{X}_n {w'_k}^T\right)\right)}{\sum_{l=1}^{K}\mathrm{P}(z_n = l|\pi')\mathrm{PL}\left(r^{(n)}|\exp\left(\bar{X}_n {w'_l}^T\right)\right)}. \quad (6)$$

Then, we can rewrite Eq. 5 into Eq. 7 below:

$$Q(W, \pi | W', \pi') = \log \mathrm{P}(\pi|\alpha) + \sum_{k=1}^{K}\log \mathrm{P}(w_k|0, \sigma^2) + \sum_{n=1}^{N}\sum_{k=1}^{K} T_{nk}\left(\log \mathrm{P}(z_n = k|\pi) + \log \mathrm{P}\left(r^{(n)}|\exp\left(\bar{X}_n {w_k}^T\right)\right)\right) \quad (7)$$

Maximization Step Eq. 7 is used to maximize the model parameters π and W.

Updating π: An update step can be written for π:

$$\pi_k = \frac{1}{\lambda}\left(\sum_{n=1}^{N} T_{nk} + \alpha - 1\right), \quad (8)$$

$$\text{where } \lambda = \sum_{n=1}^{N}\left(\sum_{k=1}^{K} T_{nk} + \alpha - 1\right) \quad (9)$$

To make sure that every π_k is positive, we accept only $\alpha > 1$, so that $\alpha - 1 = \beta > 0$ serves as a smoothing pseudo-count for each group.

Updating W: The update for groups' preference vectors w_k can be done via iterative optimization, using, for example, BFGS (Broyden-Fletcher-Goldfarb-Shanno) algorithm [22]. The function to be optimized for every $k \in \{1, 2, ..., K\}$, $w \equiv w_k$ is:

$$F(w) = -\frac{ww^T}{2\sigma^2} + \sum_{n=1}^{N}\sum_{k=1}^{K} T_{nk}\log \mathrm{PL}\left(r^{(n)}|\exp\left(\bar{X}_n w^T\right)\right) \quad (10)$$

The derivative for the d-th element $w[d]$ of the vector w can be computed as follows:

$$\frac{dF(w)}{dw[d]} = -\frac{w[d]}{\sigma^2} + \sum_{n=1}^{N}\sum_{k=1}^{K} T_{nk}\sum_{i=1}^{|\bar{X}_n|}\left(x_i[d] - \frac{\sum_{l=i}^{|\bar{X}_n|} x_l[d] e^{x_l w^T}}{\sum_{l=i}^{|\bar{X}_n|} e^{x_l w^T}}\right) \quad (11)$$

5.2 Prediction

Once the model parameters are learned, we can use the model for predictions. Here, we discuss two prediction tasks.

Group Assignment For the first prediction task, given a ranked list, predict the latent preference group that this ranked list belongs to. This task allows us to align a new ranked list to one of the learnt preference groups. To address this task, we pick the $z \in \{1, 2, ..., K\}$ that maximizes the a posteriori distribution of this assignment. Let $\bar{X}$ be an items set, and r its ranking. Given the trained model parameters, we want to maximize the following probability:

$$\begin{aligned}\mathrm{P}(z|r, \pi, W, \bar{X}) &\propto \mathrm{P}(z, r, \pi, W, \bar{X}) \\ &= \mathrm{P}(\pi|\alpha)\prod_{k=1}^{K}\mathrm{P}(w_k|0, \sigma^2)\times \\ &\quad \mathrm{P}(z|\pi)\mathrm{PL}\left(r|\exp\left(\bar{X}{w_z}^T\right)\right) \\ &\propto \mathrm{P}(z|\pi)\mathrm{PL}\left(r|\exp\left(\bar{X}{w_z}^T\right)\right) \end{aligned} \quad (12)$$

Ranking Prediction For the second prediction task, given a set of items, where a ranking for some subset of the items is known, predict the ranks of the other items. This allows us to extend the rankings to other items beyond the known ranking. To address this task, we first predict the group assignment to which the set of items belongs, based on the known subset ranking (as in the first task). Once the group assignment z^* is identified, the remaining items of $\tilde{X}$ whose rankings are not yet known are arranged into a ranked list, using the group's Plackett-Luce Regression parameter, i.e., $Y = \tilde{X}{w_{z^*}}^T$. Taking into account the Plackett-Luce model properties, greater values yield higher rank positions.

6. EXPERIMENTS

The objectives of the following experiments are two-fold. First, as PLRM both discovers the latent preference groups, as well as learns a ranking function for each group, we would like to investigate the relationship between these two objectives, particularly comparing the joint modeling approach vs. the disjoint pipeline approach. Second, since PLRM is designed for a heterogeneous ranking population, we would like to verify its applicability, particularly when compared to a baseline that assumes a homogeneous ranking population.

6.1 Datasets

We describe four datasets used in the experiments. The first two: *PubFig* and *OSR* will be our main datasets that appear in all experiments, because they have known cluster labels, which are necessary as ground truth for validating the accuracy of identifying the preference groups. In addition, we include another two datasets: *Comp* and *DCam*, with rankings but without known cluster labels, which we would use only in the second half of the experiments to evaluate ranking accuracies for heterogeneous populations.

Public Figures (PubFig). This dataset[1], described by [35], consists of 772 facial images (items) of 8 public figures ($\sim$ 100 images per person). The 8 public figures are ranked with respect to 11 physical attributes (e.g., masculine-looking, pointy nose, big lips), as listed in Table 2. Each public figure is identified by a letter[2]. Expression $A \prec B$ means that item A precedes item B in the permutation. Some items share the same rank position. The third column shows the permutation lengths possible for each attribute.

These 11 attributes are considered the ground truth preference groups, because each induces a different ranking over the 8 identities. For experiments, we construct 300 ranked lists for each attribute, for a total of 3300 ranked lists. Each list is constructed by sampling an image for each identity. The feature vector of each image is a concatenation of 512-dimensional gist descriptor and a 45-dimensional Lab color histogram. The resulting collection of ranked lists and their feature vectors (but without the ground truth labels) are pooled together. For learning, we create ten random splits, such that 90% of the ranked lists for each attribute are used for training vs. 10% for testing, and average the accuracies.

Outdoor Scene Recognition (OSR) This dataset[1], described by [35], contains 2688 scenes with different spatial envelopes from 8 categories[3]. A scene (item) is represented by its 512-dimensional gist descriptor (feature vector). The categories are organized into rankings with respect to 6 attributes (e.g., natural, open, perspective), as shown in Table 2. These attributes are considered the ground truth preference groups. As in *PubFig*, we construct 300 ranked lists for each attribute (for a total of 1800 ranked lists), and create ten random splits of 90:10 for training:testing.

Computer Survey (Comp) This marketing-related dataset[4] is in the form of surveys [42]. The subjects were asked to rate 20 personal computers (items) based on their likelihood of purchasing each computer (on a scale from 0 to 10). A computer is described by its feature vector, which indicates intrinsic characteristics of the computer (e.g., amount of RAM, CPU speed) as well as extrinsic features (e.g., hotline service availability, warranty), resulting in a total of 13 binary features. We excluded subjects with missing responses and with fewer than 5 distinct likelihood values; these were 33 out of 201 subjects. Therefore, 168 subjects were used in experiments. We induce a ranked list of computers for each subject based on the likelihood ratings.

Digital Cameras (DCam) The last dataset concerns digital cameras (items). We collected the specifications of 876 digital cameras from *www.dpreview.com* and formed feature vectors according to their specifications. These include weight, number of pixels, sensor size, body type, resulting in a total of 32 features. These cameras were manually linked to Amazon product pages (*www.amazon.com*). We used the public Amazon dataset[5] described in [31, 30]. Ranked lists among the cameras were induced from ratings given by Amazon reviewers (on a scale from 1 to 5). We retain only reviewers with at least 3 distinct rating values within the linked data, resulting in 880 ranked lists.

Table 2: Permutations on Attributes [35] (Ground Truth Rankings)

Attribute	Permutation	Length
PubFig		
Masculine-looking	S$\prec$M$\prec$Z$\prec$V$\prec$J$\prec$A$\prec$H$\prec$C	8
White	A$\prec$C$\prec$H$\prec$Z$\prec$J$\prec$S$\prec$M$\prec$V	8
Young	V$\prec$H$\prec$C$\prec$J$\prec$A$\prec$S$\prec$Z$\prec$M	8
Smiling	J$\prec$V$\prec$H$\prec${A,C}$\prec${S,Z}$\prec$M	6
Chubby	V$\prec$J$\prec$H$\prec$C$\prec$Z$\prec$M$\prec$S$\prec$A	8
Visible Forehead	J$\prec$Z$\prec$M$\prec$S$\prec${A,C,H,V}	5
Bushy Eyebrows	M$\prec$S$\prec$Z$\prec$V$\prec$H$\prec$A$\prec$C$\prec$J	8
Narrow Eyes	M$\prec$J$\prec$S$\prec$A$\prec$H$\prec$C$\prec$V$\prec$Z	8
Pointy Nose	A$\prec$C$\prec${J,M,V}$\prec$S$\prec$Z$\prec$H	6
Big Lips	H$\prec$J$\prec$V$\prec$Z$\prec$C$\prec$M$\prec$A$\prec$S	8
Round Face	H$\prec$V$\prec$J$\prec$C$\prec$Z$\prec$A$\prec$S$\prec$M	8
OSR		
Natural	T$\prec${I,S}$\prec$H$\prec${C,O,M,F}	4
Open	{T,F}$\prec${I,S}$\prec$M$\prec${H,C,O}	4
Perspective	O$\prec$C$\prec${M,F}$\prec$H$\prec$I$\prec$S$\prec$T	7
Large Objects	F$\prec${O,M}$\prec${I,S}$\prec${H,C}$\prec$T	5
Diagonal Plane	F$\prec${O,M}$\prec$C$\prec${I,S}$\prec$H$\prec$T	6
Close Depth	C$\prec$M$\prec$O$\prec${T,I,S,H,F}	4

6.2 Evaluation Tasks and Metrics

In the experiments, we evaluate the methods based on two prediction tasks that we have outlined earlier in Section 5.2.

Group Assignment The first task is to assign a ranked list to the correct preference group. This can only be evaluated on *PubFig* and *OSR*, with known preference groups.

To measure the group assignment accuracy, we compare the preference groups arrived at by a model to the ground truth. For evaluation metric, we use the Rand Index (RI), a widely used statistical measure for data clustering. This metric is defined on the space of object pairs. We want to assign two objects (ranked lists) to the same latent preference group, if and only if they belong to the same ground truth grouping. Otherwise, we want to assign them to two different latent preference groups. The former is known as true positive (TP), while the latter is known as true negative (TN). For N objects, the total number of object pairs is $N(N-1)/2$. Therefore, the Rand Index is defined as follows:

$$RI = \frac{2(TP + TN)}{N(N-1)} \tag{13}$$

Rand Index or RI ranges from 0 (worst) to 1 (best). We will express them as percentages.

Ranking The second evaluation task is to predict the ranking of items based on their features. To measure the ranking accuracy, we employ Kendall's Tau correlation coefficient. It measures how similar two ranked lists are in terms of the difference between two probabilities, namely: the probability that the observed ranked lists are in the same order versus the probability that they are not.

Given two ranked lists $A = (a_i)_{i=1}^M$ and $B = (b_i)_{i=1}^M$ in the form of permutations, we say that for $i \neq j$, a pair (a_i, b_i) is concordant with another pair (a_j, b_j) if either both $a_i \succ a_j$

[1]https://filebox.ece.vt.edu/~parikh/relative.html

[2]The 8 identities in *PubFig* are: Alex Rodriguez (A), Clive Owen (C), Hugh Laurie (H), Jared Leto (J), Miley Cyrus (M), Scarlett Johansson (S), Viggo Mortensen (V) and Zac Efron (Z).

[3]The 8 categories in *OSR* are: coast (C), forest (F), highway (H), inside-city (I), mountain (M), open-country (O), street (S) and tall-building (T).

[4]https://github.com/probml/pmtkdata/tree/master/conjointAnalysisComputerBuyers

[5]http://jmcauley.ucsd.edu/data/amazon/

and $b_i \succ b_j$, or both $a_i \prec a_j$ and $b_i \prec b_j$. Otherwise we say that the pairs are discordant. Kendall's Tau is defined as follows:

$$\tau = \frac{\#\text{ concordant pairs} - \#\text{ discordant pairs}}{\frac{1}{2}M(M-1)}. \quad (14)$$

τ can take the values between minus one and plus one. For evaluation purposes, we re-normalize the coefficient so that it yields a value from zero to one, as follows:

$$\tau^* = \frac{\tau + 1}{2} = \frac{\#\text{ concordant pairs}}{\frac{1}{2}M(M-1)}. \quad (15)$$

We use Kendall's Tau to compare the ranking produced by a method with the ground truth. Thus, higher Kendall's Tau is better. We will express the value in terms of percentages, averaging across the ranked lists in the testing set.

Where perfect rankings are known, Kendall's Tau better reflects how close an output ranking is to the perfect ranking [9]. In our datasets, all rank positions are important, and not just the top positions. For instance, in *PubFig* when ranking facial images based on a certain physical attribute, we wish to get the ranking right across the full length of the list. For that reason, Kendall's Tau is more appropriate than those favoring the top-ranked elements such as DCG.

6.3 Compare to Pipeline Approach

Here, we seek to evaluate the efficacy of the PLRM model, which joins together the tasks of discovering the preference groups as well as learning a ranking function for each group. As we look into validating both preference groups and ranking, we can use only *PubFig* and *OSR* in these experiments.

Group Assignment We first explore how well PLRM can recover the ground truth clustering structure within the data (i.e., the attributes in *PubFig* and *OSR*). The most appropriate baseline is Plackett-Luce Mixture or PLM, which is a mixture model based on Plackett-Luce that does not use the feature space representation to generalize elements beyond their identity (see Section 4). That way, we can see how PLRM's modeling of regression-based parameters based on features helps in the clustering objective. The number of latent preference groups K in both PLRM and PLM is set to the actual number of attributes in the respective datasets.

The clustering results for PLRM and PLM are shown in Table 3. Since the effect of heterogeneous rankings can most clearly be studied when the attributes are really diverse and distinct, we start with an experiment involving three such attributes. For *PubFig*, we use {Masculine-looking, Pointy Nose, Round Face}. For *OSR*, we use {Natural, Large Objects, Close Depth}. In each case, the three attributes are diverse, with the lowest cumulative Kendall's Tau-b statistics (adjusted for ties), indicating stronger disagreement in terms of the permutation among the three attributes.

Furthermore, for greater insight into results, we consider three different ways of sampling for generating ranked lists.

- In the *Random* experiment, we sample items of each identity at random for each considered attribute. For *PubFig*, both PLRM and PLM do well, achieving close to 99.8% in terms of Rand Index. In this case, the number of samples is enough to learn an appropriate ranking value for each element in the dataset with respect to the attributes (each person in *PubFig* has only about 100 images). However, for *OSR*, PLRM with 95.1% outperforms PLM with 56.1% significantly. Because it relies on identities, but not features, PLM performs worse in the case where there is insufficient ranking information for specific items, such as in *OSR*.
- In the *Exclusive* experiment, we consider three non-overlapping partitions of items, one for each attribute, from which the ranked lists are generated. In this scenario, PLRM could still learn through the feature space, getting 100% for *PubFig* and 98.7% for *OSR*. In contrast, PLM cannot learn how the same items may be ranked differently, and thus gets lower Rand Indices of 51.1% for *PubFig* and 60.5% for *OSR*. This shows the limitation of PLM when an item has not been seen across all the preference groups, which is overcome by PLRM that does not need to see the exact item if other items with similar features have been seen.
- In the *All-for-One* experiment, we first select a subset of items to rank, and then generate the ranked lists for all attributes. Next, we select a different subset of items to rank. Therefore, two ranked lists from the same attribute do not share items. Although we always see all rankings from all attributes, there is not enough information to connect different ranked lists of the same attribute. This showcases the weakness of PLM that requires to have seen cooccurrences of items, whereas PLRM that works through the feature space can still solve it, attaining 99% accuracies for both datasets, as compared to PLM's 55.7%.

Finally, we consider *all* the attributes (11 for *PubFig* and 6 for *OSR*), and show the results under the *All* columns. Overall, PLRM does a better job in clustering than PLM. For *PubFig*, PLRM's 89.4% on *PubFig* and 83.4% on *OSR* are better than PLM's results (bold indicates best results). PLM is unable to generalize from item id, while PLRM seeks ranked lists that are consistent with the ranking function.

Predicting Group Assignment with Subset Length In the previous experiments, we have assumed that we have ranked lists of sufficient length, and seek to identify the group. In some predictive scenarios, we may have a new ranked list with very few rankings for which we would like to know what its ranking function would be, in order to predict unseen rankings. We first need to identify its group assignment, in order to identify its ranking function.

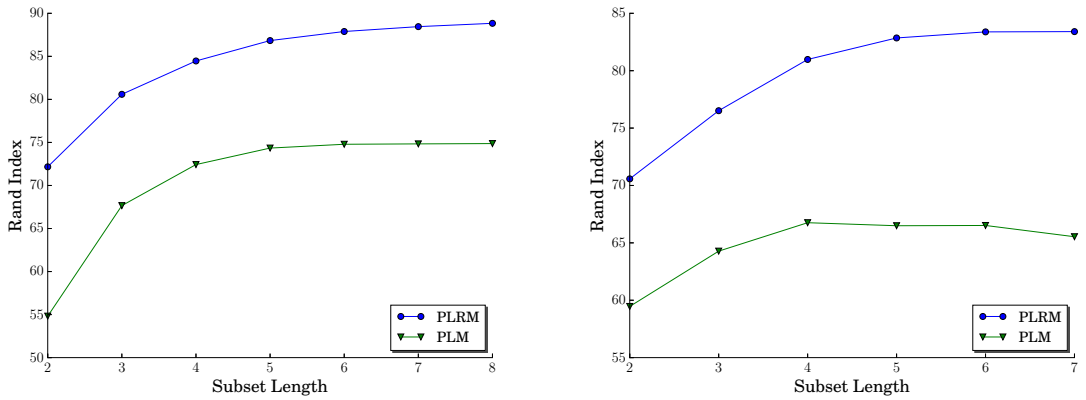
Figure 2 shows the clustering results when only a subset of the ranked list (of a specified length) is used for group assignment. This experiment is for *All* attributes with *Random* sampling. The figure shows that for both PLRM and PLM, the longer the subset length used, the more accurate is the group assignment, which is reasonable because there is more information to identify the group. In relative terms, PLRM considerably outperforms the PLM, due to the former's feature-based nature. PLM may not result in reasonable predictions for unseen items, in which case the most probable cluster according to π is chosen.

Ranking Prediction It is possible to predict unseen ranking of items if the preference group of a judge (ranked list) is known (see Section 5.2). Given a set of items, we consider the scenario when a judge is first asked to rank some subset of these items. After the initial ranking, we then predict the preference group for the judge, and determine the ranking for the rest of the items on the judge's behalf.

The previously identified baseline PLM can only perform clustering, but not ranking prediction of unseen items be-

Table 3: Group Assignment Results (Rand Index)

Method	PubFig {Masculine, Pointy Nose, Round Face} Random	PubFig {Masculine, Pointy Nose, Round Face} Exclusive	PubFig {Masculine, Pointy Nose, Round Face} All-for-One	PubFig All Random	OSR {Natural, Large Objects, Close Depth} Random	OSR {Natural, Large Objects, Close Depth} Exclusive	OSR {Natural, Large Objects, Close Depth} All-for-One	OSR All Random
PLRM	**99.8**	**100.**	**99.2**	**89.4**	**95.1**	**98.7**	**99.6**	**83.4**
PLM	**99.8**	51.1	55.7	76.3	56.1	60.5	55.7	57.9



(a) PubFig (b) OSR

Figure 2: Predicting Group Assignments Based on Subset Length

cause it does not consider features. To investigate the effects of both clustering and ranking, we consider a pipeline baseline, involving first clustering using PLM followed by learning-to-rank using PLR for each cluster (see Section 4).

Table 4 shows the results of ranking prediction when the different sampling strategies are applied, corresponding to the clustering experiments in Table 3. We reserve a subset length of 3 and 2 for *PubFig* and *OSR* respectively for first predicting the group assignment, which still leaves sufficient remaining rankings to be predicted for all attributes. Thereafter, we use the assigned group's ranking function.

In general, the ranking prediction results are consistent with the clustering results. Most of the time, when PLRM has better clustering performance, it also has better ranking performance. This is most notable for *OSR*, whereby PLRM consistently has better ranking performance than PLM+PLR across different sampling strategies. For *PubFig*, that is mostly true, with a couple of reasonable exceptions. For the three distinct attributes, in the *Random* experiment, PLRM and PLM have very similar clustering performances for reasons cited above. Therefore it is reasonable that PLRM and PLM+PLR also have very similar ranking performances (italics indicates that the difference is not statistically significant). For *All* attributes, PLRM has slightly lower ranking performance. This may be due to the fact that not all the 11 attributes in *PubFig* are distinct. As we will see shortly, the intrinsic number of preference groups is around 3 in *PubFig*, implying that some of the 11 attributes may be correlated. For *OSR*, PLRM is better.

6.4 Compare to Non-Heterogeneous Approach

We now look into the utility of PLRM in ranking scenarios, particularly comparing to methods that do not assume a heterogeneous ranking population and thus rely on a central ranking function. In addition to *PubFig* and *OSR*, in these experiments we use two additional datasets containing user opinion responses: Computer Survey (*Comp*) and Digital Cameras (*DCam*). Since we know the users who rate the products in *DCam*, we assign each user to a particular preference group. These datasets were not studied in the previous section because they lacked ground truth for clustering. However, they still allow for validation of rankings.

Number of Clusters For this comparison, we first need to determine the number of preference groups for PLRM. It is not advisable to rely on the known number of attributes. For one reason, the intrinsic number of preference groups may be different than the number of attributes. For another reason, some datasets such as *Comp* and *DCam* do not have known preference groups. Therefore, we first determine the intrinsic number of preference groups by varying K for each dataset, and measure the ranking prediction using the assigned group's ranking function. For each dataset, we try to accommodate as long a subset length for group assignment as possible, while still allowing sufficient remaining items to rank. The subset lengths are 3 for *PubFig*, 2 for *OSR*, 3 for *Comp*, and 3 to 5 for *DCam* (varying because users have rated different numbers of items).

Figure 3 shows ranking prediction quality plotted against the number of clusters for each dataset. It shows that the greatest gains come from increasing the number of clusters from 1 to 2, thereafter the performance increases slower or converges. The numbers of clusters or preference groups maximizing the ranking performance are 3 for *PubFig*, 2 for *OSR*, and 5 for *DCam*. For *Comp*, there is not much difference among different numbers of clusters, but 2 clusters are slightly better than 1; this may be an indicator that there is less heterogeneity overall for *Comp*. Subsequently, we will use these numbers to compare to the baseline.

Table 4: Ranking Results (Normalized Kendall's Tau)

Method	PubFig {Masculine, Pointy Nose, Round Face} Random	PubFig {Masculine, Pointy Nose, Round Face} Exclusive	PubFig {Masculine, Pointy Nose, Round Face} All-for-One	PubFig All Random	OSR {Natural, Large Objects, Close Depth} Random	OSR {Natural, Large Objects, Close Depth} Exclusive	OSR {Natural, Large Objects, Close Depth} All-for-One	OSR All Random
PLRM	*89.8*	**91.6**	**89.7**	85.1	**71.7**	**74.7**	**89.5**	**76.9**
PLM+PLR	**91.3**	89.2	80.6	**86.6**	63.4	66.9	86.3	66.5

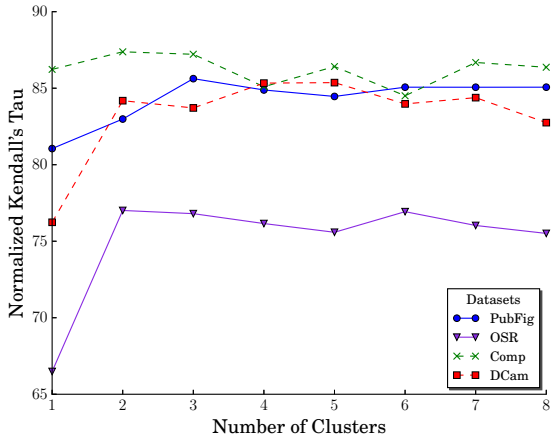


Figure 3: Ranking Prediction: PLRM with varying number of clusters K

Ranking Prediction As our focus is on validating the applicability to heterogeneous ranking population, the most appropriate baseline to PLRM in this respect is PLR (see Section 4), which is based on the same underlying Plackett-Luce regression modeling, but does not model a mixture.

Table 5 compares PLRM with the specified numbers of clusters to PLR, which effectively only has one cluster. The results show that PLRM outperforms PLR on all datasets. This outperformance is quite considerable and statistically significant for *PubFig*, *OSR*, and *DCam*. This outperformance helps to support the case that when the population has a high level of heterogeneity, a method that considers multiple latent preference groups such as PLRM have the potential to do significantly better. For a dataset that does not have a high level of heterogeneity in the first place, such as *Comp*, the improvement is rather modest.

Though PLR is a simplified version of PLRM without mixture modeling, we point out that PLR is not a weak baseline, and is actually a competitive learning to rank method in its own right. Table 6 benchmarks PLR to popular learning to rank methods, such as Coordinate Ascent [33], SVM-Rank [19], RankNet [5], ListNet [6], and RankBoost [12]. We use their implementations in RankLib[6] and SVM^{rank}[7]. Because these methods are based on very different algorithms, these are provided as a point of reference, rather than as a direct comparison. Nevertheless, Table 6 shows that PLR gets good results on the datasets. In many cases, PLR is comparable or even better. This underlines the relative strength of PLR, which in turn lends greater support to PLRM's outperformance.

[6]https://sourceforge.net/p/lemur/wiki/RankLib/
[7]https://www.cs.cornell.edu/people/tj/svm_light/

Table 5: Ranking Prediction: PLRM vs. PLR

Method	PubFig	OSR	Comp	DCam
PLRM	**85.6**	**77.0**	**87.3**	**85.4**
PLR	81.1	66.5	*86.2*	76.2

Table 6: Comparison of Learning to Rank Methods

Method	PubFig	OSR	Comp	DCam
PLR	81.1	66.5	86.2	76.2
Coordinate Ascent	75.0	58.0	81.9	75.6
SVM-Rank	78.4	67.0	86.6	71.0
RankNet	76.2	63.7	79.6	67.2
ListNet	76.6	64.4	86.8	67.9
RankBoost	78.6	61.7	83.6	58.6

Brief Comment on Running Time Our focus in this work is on effectiveness and accuracy, and not on computational efficiency. The training times are reasonable. For instance, among the learning to rank methods, PLR's training takes less than a minute. This is comparable to SVM-Rank, and faster than other learning to rank methods. In turn, the training of PLRM requires optimization of latent variables with EM algorithm. Hence, it takes more time, which increases with the required number of clusters. For instance, it takes under 30 iterations till convergence on *PubFig*, with each iteration taking a minute on average. These time measurements were conducted on a PC with Intel Core i5 CPU 3.3 GHz and 12GB of RAM running Windows OS.

Case Study To gain a sense of the nature of the clusters that PLRM learns, we show the top five features for each of the five clusters or preference groups learnt from *DCam*:

1. Pentaprism VF (viewfinder), mid-size, CMOS sensor, CCD sensor, mirrorless-style;
2. Pentaprism VF, mid-size, screen size, CMOS sensor, BSI-CMOS sensor, pentamirror VF;
3. Pentaprism VF, mid-size, Foveon X3 sensor, pentamirror VF, rangefinder-style;
4. Tunnel VF, compact, mirrorless, pentaprism VF, Foveon X3 sensor;
5. Max ISO, electronic VF, pentamirror VF, BSI-CMOS sensor, compact.

The first three groups favor mid-size cameras with pentaprism viewfinders having CMOS-like sensors. The last two preference groups give more credit to compact cameras than to mid-size cameras. The last preference group also favors cameras that can work in low-light conditions. The top features *Max ISO* (indicating maximal light sensitivity) and *BSI-CMOS Sensor* (a specific type of sensor that increases amount of light can be captured) support this observation.

7. CONCLUSION

We consider the problem of modeling ranking functions in a heterogeneous population. In such a population, there may be several latent preference groups. Each group shares a ranking function, yet across groups there are significant differences in their ranking functions. Our proposed model, Plackett-Luce Regression Mixture or PLRM, is a probabilistic graphical model that models a mixture of K latent ranking functions. Experiments show that there is value in modeling discovery of latent preference groups and ranking functions within a unified model, particularly when there is considerable heterogeneity in the data, as shown by the comparison of PLRM and the baselines: clustering-only PLM and the pipeline PLM+PLR. Moreover, PLRM also outperforms ranking-only PLR, giving credence to modeling multiple latent preference groups for heterogeneous rankings.

Acknowledgments

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its NRF Fellowship Programme (Award No. NRF-NRFF2016-07).

8. REFERENCES

[1] C. Archambeau and F. Caron. Plackett-Luce regression: A new Bayesian model for polychotomous data. In *UAI*, 2012.

[2] P. Awasthi, A. Blum, O. Sheffet, and A. Vijayaraghavan. Learning mixtures of ranking models. In *NIPS*, 2014.

[3] J. R. Bettman, M. F. Luce, and J. W. Payne. Constructive consumer choice processes. *Journal of Consumer Research*, 25(3):187–217, 1998.

[4] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, pages 324–345, 1952.

[5] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *ICML*, pages 89–96, 2005.

[6] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *ICML*, pages 129–136, 2007.

[7] F. Caron, Y. W. Teh, T. B. Murphy, et al. Bayesian nonparametric Plackett-Luce models for the analysis of preferences for college degree programmes. *The Annals of Applied Statistics*, 8(2):1145–1181, 2014.

[8] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *WWW*, 2001.

[9] B. Fernando, E. Gavves, D. Muselet, and T. Tuytelaars. Learning to rank based on subsequences. In *ICCV*, 2015.

[10] M. A. Fligner and J. S. Verducci. Distance based ranking models. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 359–369, 1986.

[11] B. Francis, R. Dittrich, and R. Hatzinger. Modeling heterogeneity in ranked responses by nonparametric maximum likelihood: How do Europeans get their scientific knowledge? *The Annals of Applied Statistics*, pages 2181–2202, 2010.

[12] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

[13] I. C. Gormley and I. T. B. Murphy. Mixed membership models for rank data: Investigating structure in Irish voting data. *Airoldi, EM, Blei, DM, Erosheva, EA and Fienberg, SE (eds.). Handbook of Mixed Membership Models and Their Applications*, 2014.

[14] I. C. Gormley and T. B. Murphy. Exploring voting blocs within the Irish electorate: A mixture modeling approach. *Journal of the American Statistical Association*, 2008.

[15] I. C. Gormley and T. B. Murphy. A mixture of experts model for rank data with applications in election studies. *The Annals of Applied Statistics*, pages 1452–1477, 2008.

[16] M. Grbovic, N. Djuric, S. Guo, and S. Vucetic. Supervised clustering of label ranking data using label preference information. *Machine Learning*, 93(2-3):191–225, 2013.

[17] J. Guiver and E. Snelson. Bayesian inference for Plackett-Luce ranking models. In *ICML*, 2009.

[18] X. Jiang, Y. Hu, and H. Li. A ranking approach to keyphrase extraction. In *SIGIR*, pages 756–757, 2009.

[19] T. Joachims. Optimizing search engines using clickthrough data. In *KDD*, pages 133–142, 2002.

[20] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.

[21] P. H. Lee and L. Philip. Mixtures of weighted distance-based models for ranking data with applications in political studies. *Computational Statistics & Data Analysis*, 56(8):2486–2500, 2012.

[22] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989.

[23] N. N. Liu, M. Zhao, and Q. Yang. Probabilistic latent preference analysis for collaborative filtering. In *CIKM*, pages 759–766, 2009.

[24] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 2009.

[25] R. D. Luce. *Individual Choice Behavior a Theoretical Analysis*. John Wiley and sons, 1959.

[26] R. D. Luce. The choice axiom after twenty years. *Journal of Mathematical Psychology*, 15(3):215–233, 1977.

[27] Y. Lv, T. Moon, P. Kolari, Z. Zheng, X. Wang, and Y. Chang. Learning to model relatedness for news recommendation. In *WWW*, pages 57–66, 2011.

[28] C. L. Mallows. Non-null ranking models. i. *Biometrika*, 44(1/2):114–130, 1957.

[29] J. I. Marden. *Analyzing and modeling rank data*. CRC Press, 1996.

[30] J. McAuley, R. Pandey, and J. Leskovec. Inferring networks of substitutable and complementary products. In *KDD*, pages 785–794, 2015.

[31] J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. Image-based recommendations on styles and substitutes. In *SIGIR*, pages 43–52, 2015.

[32] M. Meila and H. Chen. Dirichlet process mixtures of generalized Mallows models. In *UAI*, 2010.

[33] D. Metzler and W. B. Croft. Linear feature-based models for information retrieval. *Information Retrieval*, 2007.

[34] D. Metzler and T. Kanungo. Machine learned sentence selection strategies for query-biased summarization. In *SIGIR Learning to Rank Workshop*, pages 40–47, 2008.

[35] D. Parikh and K. Grauman. Relative attributes. In *ICCV*, pages 503–510. IEEE, 2011.

[36] R. L. Plackett. The analysis of permutations. *Applied Statistics*, pages 193–202, 1975.

[37] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461, 2009.

[38] Y. Shi, M. Larson, and A. Hanjalic. List-wise learning to rank with matrix factorization for collaborative filtering. In *RecSys*, pages 269–272, 2010.

[39] M. N. Volkovs, H. Larochelle, and R. S. Zemel. Learning to rank by aggregating expert preferences. In *CIKM*, 2012.

[40] M. N. Volkovs and R. S. Zemel. A flexible generative model for preference aggregation. In *WWW*, pages 479–488, 2012.

[41] M. Weimer, A. Karatzoglou, Q. V. Le, and A. Smola. Maximum margin matrix factorization for collaborative ranking. *NIPS*, pages 1–8, 2007.

[42] L. Xu, A. Huang, J. Chen, and E. Chen. Exploiting task-feature co-clusters in multi-task learning. In *AAAI*, pages 1931–1937, 2015.

[43] Z. Zhao, P. Piech, and L. Xia. Learning mixtures of Plackett-Luce models. In *ICML*, 2016.