9-19-2023

MACHINE LEARNING AND CAUSALITY FOR INTERPRETABLE AND AUTOMATED DECISION MAKING

Maria Lentini
Rowan University

MACHINE LEARNING AND CAUSALITY FOR INTERPRETABLE AND AUTOMATED DECISION MAKING

by
Maria Lentini

A Thesis

Submitted to the
Departments of Science and Mathematics
College of Science and Mathematics
In partial fulfillment of the requirement
For the degree of
Master of Science in Data Science
at
Rowan University
August 23, 2023

Thesis Chair: Umashanger Thayasivam, Ph.D., Professor, Department of Science and Mathematics

Committee Members:
Gregory Ditzler, Ph.D., Associate Professor, Department of Electrical and Computer Engineering
Shen Shyang Ho, Ph.D., Associate Professor, Department of Computer Science and Research

Dedications

I would like to dedicate this thesis to my partner and parents, whose love and support made this possible.

Acknowledgements

I would like to thank and express immense gratitude for my thesis adviser Dr. Umashanger Thayasivam, not only for his valuable lectures on multivariate data analysis, but whose considerable feedback, encouragement and belief in me made this thesis possible. Additionally, I would like to thank the other two members of the committee, Dr. Shen Shyang Ho and Dr. Ditzler, from which I have learned a tremendous amount in their respective data mining and machine learning courses, enough so as to build an early but fruitful career as a data scientist.

Abstract

Maria Lentini
MACHINE LEARNING AND CAUSALITY FOR INTERPRETABLE AND
AUTOMATED DECISION MAKING
2022-2023
Umashanger Thayasivam, Ph.D.
Master of Science in Data Science

This abstract explores two key areas in decision science: automated and interpretable decision making. In the first part, we address challenges related to sparse user interaction data and high item turnover rates in recommender systems. We introduce a novel algorithm called Multi-Veiw Interactive Collaborative Filtering (MV-ICTR) that integrates user-item ratings and contextual information, improving performance, particularly for cold-start scenarios.In the second part, we focus on Student Prescription Trees (SPTs), which are interpretable decision trees. These trees use a blackbox "teacher" model to predict counterfactuals based on observed covariates. We experiment with a Bayesian hierarchical binomial regression model as the teacher and employ statistical significance testing to control tree growth, ensuring interpretable decision trees. Overall, our research advances the field of decision science by addressing challenges in automated and interpretable decision making, offering solutions for improved performance and interpretability.

Table of Contents

Table of Contents (Continued)

List of Figures

Figure Page

List of Figures (Continued)

List of Tables

Table Page

Chapter 1

Causality and Decision Making

1.1 An Insuperable Business Dilemma

1.1.1 Causal Stories

Decision science is the branch of data science that deals with data driven policy generation. The author's interests in decision science stems from specific business requirements that developed organically during the course of the author's industrial career, and so were directly motivated by occupational demands. In particular, two aspects of decision science are investigated in two separate chapters: interpretable and automated decision making in Chapters 2 and 3, respectively.

Interpretable decision making applications are diverse, including topics such as self-driving cars, prescribing medication, and personalized pricing policies. Underlying all of these is the need to assign blame in the event of error: if the car crashes, the patient has a bad outcome, or revenue decreases we need accountability, to diagnose and pinpoint what went wrong and to facilitate remediation of the error. In black-box models, where predictions emerge as consequences of opaque calculations, this isn't possible. Hence the need for interpretability in machine learning.

Automated decision making, in contrast, is concerned with scenarios where the number decisions that need to be made are so vast and dynamic that scalability and automation become the most important factors. Classically, we think of recommender systems, which assist users in making decisions to drive outcomes such as sales or engagement. Recommender systems typically utilize collaborative filtering, which aims to measure user similarity, where users with similar item preferences are recommended similar items. Items can be diverse, ranging from YouTube videos, to online advertisements, to job postings. Furthermore, preferences are typically measured via ratings which can be explicit (i.e., a star system for movies) or implicit (i.e., clicks on a digital advertisement).

Underlying decision science is the theory of causality. For instance, to prescribe a patient medication we need a model which can infer the outcome of the prescribed treatment. Similarly, in large scale recommender systems, which prescribe items (e.g., movies, jobs, etc.) we hope to causally drive the outcomes we care about, such as sales and engagement. For this reason we spend the next few sections discussing the theory of causality.

1.1.2 Descriptive, Predictive or Causal?

Statistics and machine learning encompass three fundamental task categories: descriptive, predictive, and prescriptive. Descriptive techniques seek merely to describe “what is" quantitatively using statistics, machine learning and data visualization. However, sometimes it is advantageous to go beyond descriptive statements and instead use data to make predictions about the unseen. For instance, load forecasting electrical consumption for proper resource allocation and cost-saving. While predicting the unseen is undoubtedly advantageous, as in the aforementioned example, it sometimes imperative that we move beyond prediction and into the realm of intervention. Wind, for example, can be predicted via the motion of leaves of the tree outside our window, so inference is possible. If we would like to intentionally generate wind, however, correlation is no longer sufficient. Instead, we must understand which relationships are causal, and for that we must have a causal model of the world. Note that, when we confuse correlation with causation we can end up with bad policies such as shaking trees to generate wind. The question then is thus: how do we avoid conflating correlation and causation, and know which of the correlative variables are causal? Prescriptive models attempt to model how the world works by quantifying the effect of intervention and *prescribing* actions which influence the outcome variables we care about (e.g., clicks on an ad or the sale of a product). Causal inference does not merely forecast the unseen, but provides us with an optimal intervention strategy.

In reality, it is not always possible to perform the gold standard of causal inference, which is the randomized control trial (RCT), where the assignment mechanism is *known* a priori. In many “natural" settings the assignment mechanism is unknown. If we are interested in the causal impact of a college education on income, for instance, we cannot control the mechanism for assigning individuals to the “college" or “no-college" treatment and control groups, respectively, because no one would agree to participate, and it is probably unethical. In this thesis, we constrain ourselves to observational (i.e., non-experimental) data for which the assignment mechanism is not known and focus on addressing the large amount of applications for which RCTs are not possible. Fundamentally, at the root of prescriptive policy making is the task of performing valid causal inference, which entails the estimation of counterfactual outcomes, or so-called “would-be" or potential outcomes. If in one universe we could send a individual to college and in another not send them, the difference in their income would give use the causal effect of college for that person. If we did this for multiple people and took the average this would give us the average treatment effect (ATE). Unfortunately, we do not have separate universes for which to conduct experiments (this is known as the “hard" problem of causality). While RCTs can circumnavigate the “hard" problem and give unbiased causal estimates they are not always possible. Thus, additional tools are introduced to explain how we can infer causation in observational data.

Suppose a prescriptive policy must be made to choose from a finite set of m potential treatments such that the applied treatment $t \in [m]$ over a set of covariates $X \in \mathbb{R}^d$ yields the greatest possible benefit. W.L.O.G. we assume that maximizing Y is the primary objective:

$$t_0 = \underset{t}{\operatorname{argmax}} Y(t) \tag{1}$$

When estimating the causal influence of a treatment on an outcome variable, potential outcomes should be made conditionally independent of the treatment assignment mechanism (this is sometimes called the conditional independence assumption (CIA) or *unconfoundedness*; see Equation 2), and the probability of treatment assignment t given X_i

be nonzero (see Equation 3:

$$\left\{Y_i^{(0)}, Y_i^{(1)}\right\} \perp\!\!\!\perp T_i | X_i \tag{2}$$

$$\mathbb{P}(T_i = t | X_i) > 0 \tag{3}$$

Consider the example of an individual taking Aspirin: we know the outcome of taking Aspirin, let's say on the effect of headache, but do not know the outcome when aspirin was not taken. The unit treatment effect, indexed by i, is given by the delta between two potential outcomes corresponding to taking $(t = 0)$ and not taking $(t = 1)$ Aspirin:

$$\delta_i = Y_i(1) - Y_i(0) \tag{4}$$

Typically, we are interested in the average treatment effect (ATE), which can be found by taking the expectation on both sides of Equation 4:

$$\mathbb{E}[\delta_i] = \mathbb{E}\left[Y_i^{(1)} - Y_i^{(0)}\right] \tag{5}$$

If we find that confounding variables exist in our data (i.e., variables that influence both the probability of treatment assignment and the outcome variable) we can simply condition on them to control for their influence. Hence, a valid partitioning of units can yield unconfoundedness. We can therefore re-express equation (4) as the conditional average treatment effect (CATE):

$$\tau(x) = \mathbb{E}\left[Y_i^{(1)} - Y_i^{(0)} | X_i = x\right] \tag{6}$$

We cannot measure Equation 6 directly, since for any unit i we only observe one potential outcome. The unobserved potential outcomes (i.e., counterfactuals) must therefore be estimated. What we actually measure are the averages conditional on the treatment group:

$$\underbrace{\mathbb{E}[Y_i|T_i=1]-\mathbb{E}[Y_i|T_i=0]}_{\text{Observed difference in average outcomes}} = \underbrace{\mathbb{E}[Y_i^{(1)}|T_i=1]-\mathbb{E}[Y_i^{(0)}|T_i=1]}_{\text{Average treatment effect on the treated}} + \underbrace{\mathbb{E}[Y_i^{(0)}|T_i=1]-\mathbb{E}[Y_i^{(0)}|T_i=0]}_{\text{Selection bias}} \tag{7}$$

Equation 7, taken directly from [1], shows that the observed difference in average outcomes is the ATE with the addition of selection bias. Further note that we do not observe $\mathbb{E}[Y_i^{(0)}|T_i=1]$, the expected outcome for the treated group if untreated. In a randomized control trial unconfoundedness (Equation 2) is made true by design and therefore $\mathbb{E}[Y_i^{(0)}|T_i=0]=\mathbb{E}[Y_i^{(0)}|T_i=1]$, so that Equation 7 becomes:

$$\begin{aligned}\mathbb{E}[Y_i|T_i=1]-\mathbb{E}[Y_i|T_i=0] &= \mathbb{E}[Y_i^{(1)}|T_i=1]-\mathbb{E}[Y_i^{(0)}|T_i=0]\\ &= \mathbb{E}[Y_i^{(1)}|T_i=1]-\mathbb{E}[Y_i^{(0)}|T_i=1]\\ &= \mathbb{E}[Y_i^{(1)}-Y_i^{(0)}|T_i=1]\\ &= \mathbb{E}[Y_i^{(1)}-Y_i^{(0)}]\end{aligned} \tag{8}$$

Note that for brevity we did not condition on covariates in either Equation 7 or Equation 8, implicitly assuming CIA to hold without them. Now, as was previously discussed, it is not always possible or practical to perform an RCT, and so to mitigate the effects of selection bias in observational data we must convince ourselves that the model we have drawn up is a valid causal model. Now, aside from directly taking the difference in expectations, another familiar method of evaluating causal effects is regression, such as the equation provided below:

$$Y_i = \alpha + \beta X_i + \gamma T_i + \epsilon_i \tag{9}$$

where ϵ_i is noise. Now, if on both sides of Equation 9 we take the conditional expectations with respect to treatment and non-treatment conditioned on covariates we find:

$$\begin{aligned}\mathbb{E}[Y_i|X_i = x, T_i = 1] &= \alpha + \beta x + \gamma + \mathbb{E}[\epsilon_i|X_i = x, T_i = 1] \\ \mathbb{E}[Y_i|X_i = x, T_i = 0] &= \alpha + \beta x + \mathbb{E}[\epsilon_i|X_i = x, T_i = 0]\end{aligned} \tag{10}$$

Thus,

$$\mathbb{E}[Y_i|X_i = x, T_i = 1] - \mathbb{E}[Y_i|X_i = x, T_i = 0] = \gamma + \underbrace{\mathbb{E}[\epsilon_i|X_i = x, T_i = 1] - \mathbb{E}[\epsilon_i|X_i = x, T_i = 0]}_{\text{Selection bias}} \tag{11}$$

Comparing Equation 7 and Equation 11 we see that the selection bias terms are equivalent. If CIA holds then the selection bias in Equation 11 will disappear and the coefficient on the treatment indicator γ will give an unbiased estimate of CATE. It should be cautioned that when conditioning on covariates we could inadvertently include variables which “close" causal pathways, and therefore bias our estimates. Therefore, one of the main objectives of causal inference is convincing ourselves that the regression we have is the regression we want. The process for uncovering the regression we want is beyond the scope of this introduction. For more information the reader is referred to [2].

Of course, regression is just one of the many tools we can use to estimate causal effects. This brings us to the important propensity score theorem as described in [1] which is restated below:

Theorem 1 (The propensity score theorem)**.** *If confoundedness* $\left\{Y_i^{(0)}, Y_i^{(1)}\right\} \perp\!\!\!\perp T_i|X_i$ *holds then* $\left\{Y_i^{(0)}, Y_i^{(1)}\right\} \perp\!\!\!\perp T_i|e(X_i)$.

where the *propensity score* is defined as $e(X_i) \equiv P[T_i = 1|X_i]$. The theorem states that it is sufficient to control for the propensities themselves in place of the corresponding covariates. A proof is provided below:

Proof.

$$\begin{aligned}
P[T_i = 1|Y_i^{(j)}, e(X_i)] &= \mathbb{E}[T_i|Y_i^{(j)}, e(X_i)] \\
&= \mathbb{E}\{\mathbb{E}[T_i|Y_i^{(j)}, e(X_i), X_i]|Y_i^{(j)}, e(X_i)\} \\
&= \mathbb{E}\{\mathbb{E}[T_i|Y_i^{(j)}, X_i]|Y_i^{(j)}, e(X_i)\} \\
&= \mathbb{E}\{\mathbb{E}[T_i|X_i]|Y_i^{(j)}, e(X_i)\} \text{ (unconfoundedness.)} \qquad (12) \\
&= \mathbb{E}\{P[T_i = 1|X_i]|Y_i^{(j)}, e(X_i)\} \\
&= \mathbb{E}\{e(X_i)|Y_i^{(j)}, e(X_i)\} \\
&= e(X_i)
\end{aligned}$$

□

The consequence of unconfoundedness and Theorem 1 is the following:

$$\tau(x) = \mathbb{E}\left[Y_i\left(\frac{T_i}{e(x)} - \frac{1-T_i}{1-e(x)}\right)|X_i = x\right] \tag{13}$$

Thus, inferring the propensity score function $e(X_i)$ allows us to obtain unbiased estimates of the CATE by conditioning on $e(x)$ in place of covariates X_i. Propensity $e(X_i)$ can be estimated using a number of machine learning models including logistic regression, neural networks, and so on. Equation 13 is typically referred to as inverse propensity weighting (IPW) since we inversely weight treatments indicators by their corresponding propensities to obtain unbiased estimates of the treatment effect.

We have now touched briefly on both regression and inverse propensity weighting. There is a third category known as “matching", which works by matching every unit in the control group with a unit in the treatment group via covariates. Similar to regression and IPW, when CIA holds matching can be used to obtain unbiased estimates of CATE. According to [1], because regression and matching both control for covariates and can be viewed as weighted matching estimators the differences between them are unlikely to be relevant empirically. As a final note, it is not uncommon for models to include unobserved confounders, such as “ability" on income in the “college" vs “no-college" example, but techniques for addressing this problem, such as instrumental variables, are beyond the scope of this introduction.

1.2 Literature Review

1.2.1 Interpretable Policy Trees

A class of functions which recursively partition the covariate space along axis-aligned partitions are known as decision trees, which in general can be categorical or regressive [3]. The tree begins by associating all data points with a root node and proceeds by searching the sample space for the “best" datapoint for which to split the data. The data is then partitioned into a pair of disjoint sets, and so on until a stopping criteria is met. The classification of a datapoint belonging to a leaf node is given by the preponderance of a class in the case of categorical data, or the average in the case of continuous data. In general, the criteria for splitting can vary from model to model. In the case of classification, the partition is commonly chosen to minimize the Gini index, entropy or classification error; in regression it is typically the mean square error (MSE). Decision trees have the added benefit that they are interpreteble, meaning that it is easy for a human to understand the decision making process of the model.

While desirable, the interpretability of a DT comes at the expense of instability, whereby small perturbations in the training dataset can result in drastically different models. This variability can be mitigated without increasing prediction bias through sub-sampling or bootstrapping methods, e.g., random forests [4] or gradient boosting in [5], where a series of “weak" (i.e., shallow) decision trees are constructed in sequence and averaged together. The improved stability (i.e., decreased variance) reduces interpretability, as it is more difficult to interpret the averaging of several base models then it is a single one.

In the pursuit of overcoming instability some authors have considered the task of constructing globally optimized decision trees, as in Optimal Classification Trees [6] and

Optimal Policy Trees [7], which treat the problem of constructing an optimal decision tree, known to be NP-complete [8], as one of mixed integer optimization. In contrast, [9] has argued that a greedy approach is more amenable to interpretability in that it mirrors decision making in humans, and therefore is preferable. This argument presupposes that human decision making processes - arguably short-term and myopic - are advantageous in all circumstances. Nevertheless, the appeal of having a single globally optimized decision tree is that we maintain both the interpretability of single tree and performance competitive with black-box models.

There are other notable algorithms which have addressed the problem of interpretable personalization, including causal trees [10], which search for and exploit heterogeneity in treatment effects, utilizing Breiman's decision tree and Rubin's potential outcomes framework; and personalization trees [11] which utilize a "personalization" impurity designed to mitigate risk in the treatment assignment policy. In this thesis, we focus on a variant of the Student Prescription Tree (SPT) from [9], a DT approach based model distillation.

SPTs are interpretable policy trees which utilize a blackbox "teacher" model to predict counterfactuals on the basis of observed covariates and a interpretable decision tree (DT) "student" model which sorts datapoints into axis aligned partitions such that the sum over predicted counterfactuals outcome variables within each partition is optimized greedly. In this thesis, we maintain this framework but experiment with a Bayesian hierarchical binomial regression model as the teacher, and require all gains acquired from DT splits to be statistically significant given the posterior distributions over outcome variables from any two disjoint partitions of the data when compared to their unity. This setup allows us to control the uncertainty (i.e., "risk") associated with a deepening of the complexity (i.e., "depth") of the student policy, and gives us natural tree growth stopping criteria. In general, greater uncertainty will result in shallower tree policies.

1.2.2 Collaborative Filtering

Recommender systems are meant to solve the information overload problem faced by users in today's online landscape, where providing the right content at the right time saves the user time, energy, increases user engagement and improves the user experience.

Following [12], we follow the convention of subdividing recommender systems into three broad classes based on the types of data they utilize: 1) models which are based on rating or implicit interaction data are termed *collaborative filtering* (CF) models [13], 2) those which utilize in addition to ratings user or item content (e.g., a user's age, personality, gender, etc.; an item's category, image, knowledge graph, etc.) are termed *content-enriched* models [14], and 3) those which utilize context, defined as things external to both user and item (e.g., time, location, weather, etc.) are termed *context-aware* models. Models which combine any of the above methods are termed *hybrid* models.

CF recommends items from users with similar preferences, whilst content-enriched models recommend either similar items or items from similar users. Content-enriched models will rely on item (user) features to measure similarity between items (users); we refer to these models as item-based and user-based, respectively. Item-based models suffer and are limited in that they can only recommend items similar to those a user has already rated, and does not know which items to recommend users who have not yet rated anything. Similarly, user-based models can only recommend items from similar users, and cannot recommend items for which there are not yet any ratings. CF suffers from both of these defects; that is, it does not know how to recommend items to new users nor users to new items. This is known as the cold-start problem, which is the problem of what to recommend users, and to whom to recommend items, which have no ratings.

Note that the features of contextual variables are neither user or item-specific, but apply equally to both, because they describe variable aspects of the environment in which users and items interact. User and item feature sets may overlap (e.g., the skills required for a job may also be possessed by a user), but it is not necessarily so. We can, therefore, define contextual features as those which are necessarily common among users and items.

CF and content-based models are *stationary*, meaning the underlying generative distributions are assumed to remained fixed in time. MABs can address the non-stationary problem by weighting older information less than newer information [15]. Furthermore, they can be adapted to associate contextual variables with actions; these are the so-called associative or contextual bandits [16]. In either the stationary or non-stationary case, the actions (i.e., interventions) associated with contextual or non-contextual bandits are assumed to only influence immediate reward. For example, if I pull the lever of a slot machine a reward will be observed (I will win or not win) and it is safe to assume that the act of pulling the lever will not influence the underlying state distribution of the slot machine itself: my probability of winning the next round is the same as before, or evolves independent of the actions I choose. Formally, the transition probability $\Pr(z_{t+1}|z_t, i_t) = \Pr(z_{t+1}|z_t)$, where the probability of the state z_{t+1} at step $t+1$ is conditionally independent of the arm i_t selected at time t: $z_{t+1} \perp\!\!\!\perp i_t|z_t$. But this is nothing more than the CIA Equation 19, with the outcome variable replaced with z_{t+1}. Contextual bandits are intermediary between MABs and full-on reinforcement learning. Unlike MABs or contextual bandits, the actions produced by reinforcement learning (RF) algorithms are assumed to affect not only the immediate reward, but also the next state. For instance, in playing chess the act of moving a pawn changes the environment, and therefore, the probability of obtaining a reward. In the context of recommender systems, the primary feedback of interest is that of user rated items, which can be explicitly obtained (e.g., thumbs up or down, one-to-five stars, etc.) or implicitly computed (e.g., watched the video to the end, clicked "apply to job", etc.).

"Interactive" CF systems are systems in which the recommendation policy is constantly being updated to reflect users' feedback, and thereby evolve in accordance with shifting trends, tastes and attitudes [17], [12]. In this thesis, we utilize non-contextual MABs, assume stationarity, and focus on developing an interactive hybrid model which combines CF and content-based methods. Specifically, we utilize *BayesMatch* (BM) [18] and *inter-*

active collaborative topic regression (ICTR) [19] to model feature and item-dependencies within a single framework. BM is a multi-view probabilistic model for clustering users and items via user-specific, item-specific and common features. ICTR combines probabilistic matrix factorization (PMF) [20] with multi-armed bandits (MABs) to generate explore-exploit optimized recommendations. Albeit not the first to do so ([17], [21], [22]), ICTR explicitly models item-dependencies, clustering items according to how they are preferred by users. Although technically solving the cold start problem, cold start recommendations in ICTR are nevertheless non-optimal, because we cannot not know beforehand which item-clusters a new user will be partial towards. It is therefore hypothesized that ICTR can be improved upon via the integration of BM feature-dependencies for cold and early-start recommendations.

Chapter 2

Interpretable Decision Making with Decision Trees

2.1 Introduction

In this chapter we seek to address the growing need for data driven pricing policies in the business context. Often times policy makers are left with gut-feel pricing decisions which may not be optimal. When policy makers seek policy prescriptions from opaque, black-box machine learning models they often run into a fundamental dilemma: how to convince a broader stakeholder buy-in? With interpretable prescriptive models, policy makers can understand how the decisions are being made, agree or disagree with them, explain them to their colleagues, and bring transparency to the decision-making logic. In this way we understand that interpretability is not only a "nice-to-have" but a necessary and vital component in applications of machine learning in the business decision making context. Our research in this arena focuses on optimal pricing for revenue maximization most similar to [9], Personalization Trees [11], Causal Forests [10] and Optimal Prescription Trees [6].

2.2 Problem Definition

In this section we consider the approach of the student presciption tree (SPT) [9], where the task is to construct an interpretable decision tree which personalizes price so as to maximize revenue. The *revenue maximization criterion* is given by:

$$\mathcal{R}(S_l) = \max_p \sum_{i \in S_l} p \hat{f}(x_i, p) \tag{14}$$

where p is price, $x_i \in \mathcal{R}^N$ are the features corresponding to datapoint i, $\hat{f}(x_i, p)$ is a function which estimates the purchase probability, and S_l represents the class of all datapoints associated with node l. The datapoint i can be thought of as an instance of a purchasing decision, where x_i can include characteristics from both the potential buyer, product or environment. At each node we consider partitioning the data into two sets:

$$S_1(j,s) = \{i \in [n] | x_{i,j} \leq s\} \tag{15}$$

$$S_2(j,s) = \{i \in [n] | x_{i,j} > s\} \tag{16}$$

where the j^{th} feature of observation i is given by $x_{i,j}$. The splits should be chosen so as to greedily maximize revenue at every split, which can be accomplished by finding the feature j and observation i which maximizes the sum of predicted revenue over the children: $\max_{j,s}[\mathcal{R}(S_1(j,s)) + \mathcal{R}(S_1(j,s))]$.

Suppose we are given a discrete set of price points $\mathcal{P} = \{p_1, p_2, ..., p_m\}$ and are asked to assign the best price to each individual. Then, the revenue for datapoint i and price k is given by $r_{i,k} = p_k \hat{f}(x_i, p_k), k \in [m]$. The revenue associated with a node is therefore given by:

$$\mathcal{R}_m(S_l) = \max_{k \in [m]} \sum_{i \in S_l} r_{i,k} \tag{17}$$

Before training the decision tree, a so-called “teacher" model $\hat{f}$ is trained to learn demand from the training data. In the original paper on SPT [9] the gradient boosted ensemble lightGBM was used as the teacher.

2.3 Risk-Controlled Revenue Personalization

When increasing the depth of a SPT, revenue - as predicted by the teacher model - must necessarily *not* decrease over the train set, however this does not bar the possibility of revenue decreasing over the test dataset. When this occurs we have *overfit* the model. Given the teacher model, associated with every prediction is a degree of uncertainty. It is precisely this uncertainty that encompasses the focus of this proposal, where our goal to model prediction uncertainty and exploit this knowledge to make risk-controlled decisions. This prevents the accepting of a prediction which promises high reward, but in reality comes with the extra baggage of high uncertainty. Areas of high uncertainty are typically associated with unexplored areas, such as product prices significantly higher or lower than the mean over a given distribution of price assignments.

In SPT, we look for the split that maximizes expected revenue. The modified version that we propose, a “risk-controlled" SPT (RC-SPT) will do the same, but while imposing an additional constraint on the splitting criteria, requiring $\Pr(R^* > R) \geq a$, such that the proposed revenue associated with the split is greater than that of its unity (see Algorithm 4). In other words, we only defer to the axis aligned splits associated with predicted revenue R^* if we have some specified degree of certainty that revenue will increase. For code, the GitHub associated with this work can be found in the footnote [1].

Algorithm 1 Risk-Controlled SPT - Student Fit

```
procedure FIT($X$)
    $R^* \leftarrow 0$
    root_node["revenue"] $\leftarrow \emptyset$
    for $P$ in prices do
        $R^* \leftarrow$ GET_POSTERIOR($[n], P$)
        if IS_RISK_CONTROLLED($R^*, R, a$) then
            root_node["revenue"] $\leftarrow \bar{R}^*$
        end if
    end for
    Split(root_node)
end procedure
```

[1] Risk-Controlled SPTs: https://github.com/lentin26/risk-averse-trees

Algorithm 2 Risk-Controlled SPT - Recursive Split

```
procedure SPLIT(parent)
    Initialize_Node(parent)
    $R \leftarrow$ parent["revenue"]
    for $i$ in parent[datapoints] do
        for $j$ in features do
            $s \leftarrow x_{ij}$
            $S_1 \leftarrow \{i^* \in [n] \,|x_{i^*j} \leq s\}$
            $S_2 \leftarrow \{i^* \in [n] \,|x_{i^*j} > s\}$
            for $P$ in prices do
                $R_1 \leftarrow$ GET_POSTERIOR($S_1, P$)
                $R_2 \leftarrow$ GET_POSTERIOR($S_2, P$)
                $R^* \leftarrow R_1 + R_2$
                if IS_RISK_CONTROLLED($R^*, R, a$) then
                    parent["revenue"] $\leftarrow R^*$
                    parent["price"] $\leftarrow P$
                    parent["split value"] $\leftarrow s$
                    parent["split feature"] $\leftarrow j$
                    parent["left child"][datapoints] $\leftarrow S_1$
                    parent["right child"][datapoints] $\leftarrow S_2$
                    parent["children"] $\leftarrow$ True
                end if
            end for
        end for
    end for
    if parent[depth] < max_depth - 1 then
        Split(parent[child])
    end if
end procedure
```

Algorithm 3 Risk-Controlled SPT - Initialize Nodes

```
procedure INITIALIZE_NODE(parent)
    parent["children"] $\leftarrow$ False
    parent["split value"] $\leftarrow$ None
    parent["split feature"] $\leftarrow$ None
    parent["left child"] $\leftarrow \emptyset$
    parent["right child"] $\leftarrow \emptyset$
    parent["left child"][depth] $\leftarrow$ parent["depth"] + 1
    parent["right child]"[depth] $\leftarrow$ parent["depth"] + 1
end procedure
```

Algorithm 4 Risk-Controlled SPT - Get Posterior Revenue Prediction

```
procedure GET_POSTERIOR($S, P$)
    $R^* \leftarrow \texttt{SUM}(\alpha^T X[S] + \beta^T X[S]P)$
    return $R^*$
end procedure
```

We can model uncertainty by training a Bayesian hierarchical model, such as the following:

$$\begin{aligned}
&A \sim \text{Binomial}(y|n, \theta) \\
&\text{logit}(\theta) = \alpha^T X_i + \beta^T X_i p \\
&\alpha \sim \mathcal{N}(0, \sigma_\alpha) \\
&\beta \sim \mathcal{N}(0, \sigma_\beta) \\
&\sigma_\alpha \sim \text{IG}(1, 1) \\
&\sigma_\beta \sim \text{IG}(1, 1)
\end{aligned} \tag{18}$$

where N is the total number of purchasing events (in general, the same individual can purchase the same product multiple times under different contextual circumstances) and $X_i \in \mathbb{R}^{D+1}$ are the covariates associated with the i^{th} purchase, where we let $X_0 = 1$ to capture the intercept.

It should be mentioned that the conditional independence assumption (CIA), also known as *unconfoundedness*, is true for some of the data models presented in this section. CIA says the following:

$$\left\{Y_i^{(0)}, Y_i^{(1)}\right\} \perp\!\!\!\perp T_i | X_i \tag{19}$$

This says that potential outcomes $Y_i^{(0)}, Y_i^{(1)}$ are independent of treatment assignment T_i conditional on covariates X_i. The binary outcome variable of concern to us is the purchasing decision made by a customer to buy a product ($Y_i = 1$) given some covariates X_i. Throughout the remainder of this chapter, we assume the generative model structure for item purchases to be a logit:

$$Y^* = g(X) + h(X)p + \epsilon \tag{20}$$

where,

$$Y = \begin{cases} 1 & Y^* > 0 \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

Note that as $h(X) \to 0$ the customer in the purchasing scenario becomes infinitely price insensitive, where the expected purchasing decision converges to a dependency on the sign of the intercept $g(X)$ alone, which is nothing more than a transformation g on a concatenation of user, item and contextual features X. Now, for price sensitive customers with $h(x) < 0$ there always exists a price threshold for the customer i given by $p_i^{\max}$ such that the revenue from the sale is:

$$R_i(p) = p \cdot \delta(p \leq p_i^{\max}) \tag{22}$$

where,

$$\delta(p_m \leq p_i^{\max}) = \begin{cases} 1 & p_m \leq p_i^{\max} \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

is the kronecker-delta, with $i \in [N]$ and $m \in [M]$ where N is the number of purchasing decisions and M are the number of discrete price options. Thus, Equation 22 is a $(N \times M)$ revenue matrix with components $R_{i,m} = \delta(p_m \leq p_i^{\max})$. Referencing Equation 20, the maximum price allowable for a sales conversion with respect to individual i, is given when $Y^* = 0$. Therefore:

$$p_i^{\max} = \frac{(-g(x_i) - \epsilon)}{h(x_i)} \tag{24}$$

Of course, the ground truth revenue matrix is typically not known in advance, and so must be deduced. This is entirely the purpose of the teacher model: to estimate $R_{i,m}$. The teacher model is used to predict the expected revenue in a purchasing instance i with price p_m, given by:

$$\mathbb{E}[R_i(p)] = p \cdot \Pr(Y_i = 1 | T_i = p) \tag{25}$$

The estimated revenue matrix is therefore given by $\hat{R}_{i,m} = \mathbb{E}[R_i(p_m)]$. The ground truth and estimated revenue matrices are provided below for ease of comparison:

$$R_{i,m} = \delta(p_m \leq p_i^{\max}) \qquad \text{(ground-truth revenue matrix)}$$

$$\hat{R}_{i,m} = p_m \cdot \underbrace{\Pr(Y_i = 1|T_i = p_m)}_{\text{Teacher Model}} \qquad \text{(teacher-estimated revenue matrix)}$$

Thus, given a generative model such as Equation 20 we can directly compare the optimal tree (training via the ground truth revenue matrix) against SPT and RC-SPT (training via the teacher-estimated revenue matrices). Now, armed with the above, the optimal revenue for a partition of the data S_l is given by:

$$\mathcal{R}_m(S_l) = \sum_{i \in S_l} \delta(p_m \leq p_i^{\max}) \tag{26}$$

$$\approx \sum_{i \in S_l} p_m \cdot \Pr(Y_i = 1|T_i = p_m) \tag{27}$$

Of course, total revenue is just the sum of revenue over all partitions. Given a set of datapoints associated with a parent node, we split greedily into left and right children according to the following:

$$\underset{(m_1, m_2, j, s)}{\operatorname{argmax}} \left[\mathcal{R}_{m_1}(S_1(j,s)) + \mathcal{R}_{m_2}(S_2(j,s))\right] \tag{28}$$

In general $m_1 \neq m_2$, since assigning the same price to both left and right nodes does not further personalize pricing. This condition is implicitly contained in the increasing revenue criteria $\Pr(R^* > R) \geq a$, as $m_1 = m_2$ necessarily implies that the combined posterior of left and right child nodes are equivalent to the posterior of the parent $R^* = R$, and hence for $a > 0$ the revenue increasing criteria is violated. Thus, the optimal prices assigned to left and right child nodes will always differ.

In RC-SPT, we use a hierarchical binomial regression (Equation 18) to estimate the full posterior distribution of the purchase probability $\hat{\theta} = \Pr(Y_i = 1|T_i = p)$ and impose risk controls on the conditions for splitting. The idea here is to avoid generating policies which accept high theoretical reward expectations with high uncertainties. Therefore, we expect the risk-controlled version to outperform vanilla SPT in regions where uncertainty is high and overfitting likely.

Algorithm 5 Risk-Controlled SPT - Evaluate Risk

```
procedure IS_RISK_CONTROLLED($R^*, R, a$)
    $x \leftarrow R^* - R$
    conf $\leftarrow$ SUM($\{x > 0\}$) / $|x| \geq a$
    return conf
end procedure
```

2.4 Experimental Results

2.4.1 Synthetic Datasets

Described below are the generative datasets similar to those utilized in [9] over which the models are evaluated, all of which all obey the logit model Equation 20:

- Dataset 1: linear probit model with no confounding: $G(X) = X_0$, $h(X) = -1$, $X \sim N(5, I_2)$ and $P \sim N(5, 1)$.

- Dataset 2: higher dimension probit model with sparse linear interaction $g(X) = 5$,

$h(X) = -1.5(X'\beta)$, $X_{i_{i=1}}^{2}0$, $\{\beta_i\}_{i=6}^{5}$, $\epsilon_i \sim N(0,1)$, $P_i \sim N(5,2)$, $\{\beta_i\}_{6}^{20} = 0$, where the purchase probability is only dependent on the first 5 features.

- Dataset 3: probit model with step interaction: $g(X) = 5$, $h(X) = -1.2, \mathbb{1}\{X_0 \leq 1\} - \mathbb{1}\{-1 \leq X0 \leq 0\} - 0.9\mathbb{1}\{0 \leq X_0 \leq\} - 0.8\mathbb{1}\{1 \leq X_0\}$.
- Dataset 4: probit model with multi-dimensional step interaction: $g(x) = 5, h(X) = -1.25\mathbb{1}\{X_0 \leq 1\} - 1.1\mathbb{1}\{-1 \leq X_0 < 0\} - 0.75\mathbb{1}\{1 \leq X_0\} - 0.1\mathbb{1}\{X_1 < 0\} + 0.1\mathbb{1}\{X_1 > 0\}, P \sim N(X_0, 2)$.
- Dataset 5: linear probit model with observed confounding: $G(X) = X_0, h(X) = -1, X \sim N(5, I_2)$
- Dataset 6: probit model with non-linear interaction: $g(X) = 4|X_0 + X_1|$, $h(X) = -|X_0 + X_1|$.

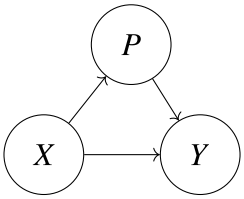


Figure 1. Directed acyclic graph (DAG) for synthetic dataset 5 shows the dependency of the outcome Y on price P and covariates X and also the confounding dependence of X on P.

We let $(X_0, X_1) \sim N(0, I_2), P \sim N(X_0 + 5, 2), \epsilon \sim N(0,1)$ unless otherwise noted. Datasets 1 and 5 are commonly used to model linear demand in the pricing literature. Note that dataset 5 is confounded apropos of a price dependency on the covariates X_0.

Dataset 5 makes sense heuristically, since it is easy to imagine a situation where demand and price both vary according to covariates. As an example, consider strawberries in a grocery store, where price can depend on the attributes of the strawberries themselves, (e.g., organic vs non-organic), while demand can depend both on price, strawberry, individual person attributes and context (e.g., buying strawberries in summer vs winter). Intuitively, an individual on average will be more likely to pay higher prices for a higher quality strawberries, and so “willingness to pay" Y can in general depend both on the strawberry quality X and price P. Individual characteristics, also contained in X, such as age, can also influence Y on the purchase of, say, candy. In other words, confounding in purchasing decisions are near ubiquitous. Note that, the conditional independence assumption (CIA) Equation 19 between the treatment assignment variable (m corresponding to price p_m) is not independent of the outcome variable Y unless we condition on all confounding covariates. Of course, we only know the confounding relationships in dataset 5 because we know the underlying generative distribution. The confounded relationship in dataset 5 is shown graphically as a directed acyclic graph (DAG) in Figure 1, which are often used in the causal inference literature for visualizing random variable dependencies.

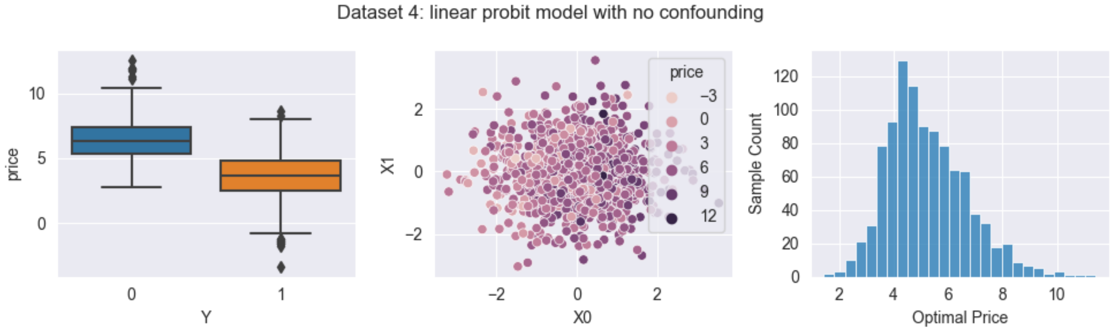
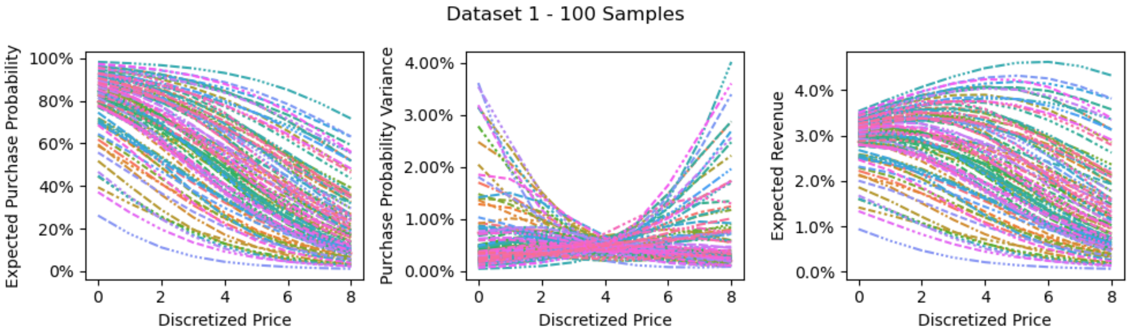


Figure 2. 1000 samples generated from dataset 4. From the leftmost figure, we see that instances of user-product purchase considerations are less likely to result in a purchase ($Y = 1$) if the prices are higher. The center figure demonstrates that price scales as a function of X_0 only, suggesting that X_0 is a price-influencing (i.e., confounding) product feature (e.g., organic vs non-organic bananas). On the right, we see the distribution of the optimal prices provided by Equation 24.

Referencing Figure 3, where the binomial regression model Equation 18 was trained on a dataset of 100 generated points, we observed that for all datapoints the expected purchase probability decreased with price, with some datapoints more price sensitive than others. On the rightmost plot, we found that all datapoints had different optimal prices. The objective of an SPT is to divide these datapoints into axis-aligned partitions so as to maximize the expected revenue as predicted by the teacher model. Finally, noting that most datapoints live in the neighborhood of the expected price $m = 5$, we observed from the center plot uncertainty in purchase probability to be at a minimum at this location, and increasing in either direction. This tells us that uncertainty is greater in regions where data is fewer, as expected.



.

Figure 3. Sample size 100 from dataset 1, shows plot of predicted individual purchase probabilities (left) with variances (center) and expected revenues (right) against price via binomial regression, eqn Equation 18

2.4.2 Experimental Setup

In this section, we record our observations for RC-SPT experiments, including how the average of leaf node depths varies as a function of the size of the sample, and set the maximum tree depth (if the stopping criteria is not reached before then) at 5. We took 1000 samples using MCMC rejection sampling with a target acceptance rate of 95% to approximate the revenue posterior for each discretized price point. In every instance, we split the generated data 50/50 for training and testing and choose $a = 0.95$ as the required probability of revenue increasing for each split.

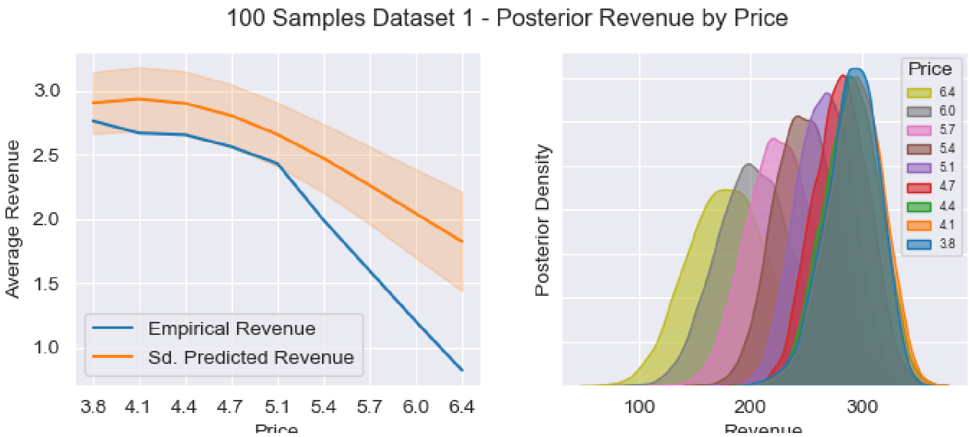


Figure 4. Left, synthetic dataset 1 with size $N = 100$, binomial regression prediction vs empirical average revenue plot, with orange shading representing 1 standard deviation. Right, sample size 100 from dataset 1, shows the posterior revenue distributions for each discretized price point summed over all 100 generated datapoints.

For more insight into what the RC-SPT is doing, consider right-hand plot of Figure 4, which gives us a sneak-peak into how the algorithm “sees" the data inside of a node. What RC-SPT effectively does is look at the posteriors at each price p_m for any two partitions of the data from the parent node, and chooses prices such their combined posterior revenue is greater than that of their parent with probability a. The left-hand plot shows the empirical revenue vs that predicted by the hierarchical regression, with shading given by 1 standard deviation. In this case, with a linear generative distribution, the posterior revenue fits the empirical data fairly well.

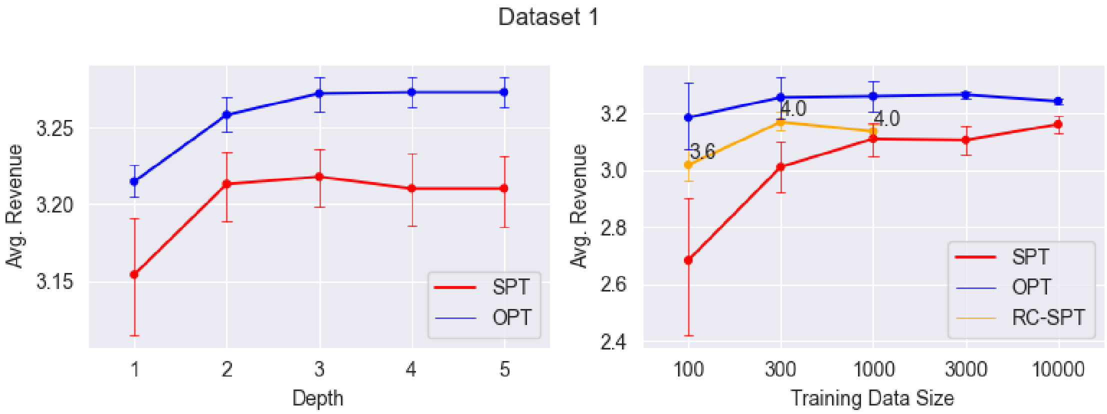


Figure 5. Experiments on dataset 1. Top shows SPT vs the optimal prescription tree (OPT) average revenue over depth. The bottom plot show SPT, OPT and RC-SPT average revenue over training dataset size. The labels on the RC-SPT (orange) curve show the average maximum depth for each of the RC-SPT trees grown (all SPT and OPT trees are set to a max depth 3). Note that RC-SPT was only trialed 5 times for training data size 3000 due to time constraints. In all other instances the number of trials conducted was 10.

2.5 Results and Discussion

Referencing Figure 5, we see that RC-SPT outperformed SPT significantly when the training size was small, specifically at sizes 100 and 300. This could be due to the fact that SPT was set to have a max depth of 5 resulting in overfitting, evidenced by the observation that the maximum depth for RC-SPT never exceeded 4 even when the training data size was 1000. Larger dataset sizes were not measured for RC-SPT due to its slow training speed.

Table 1

Varying Size Results

Dataset	100			**300**			**1000**		
	RCS	SPT	OPT	RCS	SPT	OPT	RCS	SPT	OPT
1	**2.97**	2.68	3.18	**3.19**	3.01	3.25	3.12	3.11	3.26
2	0.64	0.75	0.81	0.61	0.69	0.68	0.6	**0.69**	0.66
3	0.31	**2.93**	3.32	0.19	**3.2**	3.43	0.15	**3.21**	3.38
4	0.46	**2.84**	3.37	0.53	**3.07**	3.38	0.43	**3.23**	3.41
5	3	2.91	3.05	**3.11**	3.08	3.02	**3.13**	3.08	3
6	0.41	**2.05**	2.21	0.41	**2.13**	2.32	0.43	**2.32**	2.39

Table 1 summarizes all experimental results, showing the average of expected revenues obtained over 10 trials, with boldface signifying statistically significant improvement between RC-SPT (shortened to RCS) and SPT. RC-SPT was found to significantly underperform SPT on datsets 3 (step-interaction), 4 (multi-dimensional step interaction) and 6 (non-linear interaction), but was found to perform the same or better on datasets 1 (linear no confounding), 2 (high-dimensional linear) and 5 (linear with confounding), with the exception of dataset 2 with 1000 training set points. This is likely due to the fact that the hierarchical binomial regression model is explicitly linear between the logit of purchase probability and covariates with price, and therefore less amenable to fitting non-linear relationships.

2.5.1 Conclusions

Despite being superior only in linear circumstances, RC-SPT improved on SPT by providing an explainable stopping criteria which increased overall interpretability. Without stakeholder buy-in, models such as these cannot go into production, and therefore the gain in interpretability we deem as a valuable addition to the original SPT.

2.5.2 Future Work

RC-SPTs provide a stopping criteria according to the probability of increasing revenue via splitting, which automates the task of learning risk-controlled tree structures. RC-SPT has a complexity of $O(M \times N \times D)$, which is much slower than current state of the art tree algorithms, while the rejection sampling used to obtain posteriors is much slower than efficient teacher models like XGBoost and LightBGM. In future work, therefore, we would like to improve the model complexity and investigate the performance of rejection sampling against other MCMC methods such as variational inference (VI) or Gibbs sampling. Furthermore, the experiments conducted in this chapter can be easily extended beyond synthetic datasets to real-word ones and hence used to evaluate real-world efficacy. And lastly, although a linear model was used as the counterfactual predictor in this chapter, more flexible, non-linear Bayesian models could also be tried.

Chapter 3

Automated Decision Making with Collaborative Filtering

3.1 Introduction

In many scenarios, recommender system user interaction data such as clicks or ratings is sparse, and item turnover rates (e.g., new articles, job postings) high. Given this, the integration of contextual “side" information in addition to user-item ratings is highly desirable. Whilst there are algorithms that can handle both rating and contextual data simultaneously, these algorithms are typically limited to making only in-sample recommendations, suffer from the curse of dimensionality, and do not incorporate multi-armed bandit (MAB) policies for long-term cumulative reward optimization. We propose multi-view interactive topic regression (MV-ICTR) a novel partially online latent factor recommender algorithm that incorporates both rating and contextual information to model item-specific feature dependencies and users' personal preferences simultaneously, with multi-armed bandit policies for continued online personalization. The result is significantly increased performance on datasets with high percentages of cold-start users and items.

3.2 Interactive Collaborative Topic Regression

3.2.1 Latent Dirichlet Allocation

Interactive Collaborative Topic Regression (ICTR) works by learning lower dimensional latent vector representation of users and items based on rating data alone. The user vectors u_m are multinomial, and are therefore constrained to a hyperplane in $\mathbb{R}^{D-1}$ space, since $\sum_{k=1}^{D} u_{m,k} = 1$ for all users $m \in [M]$, whereas the item vectors are multivariate normal and therefore can exist outside of the user hyperplane anywhere in D-dimensional real space.

To understand ICTR, it is best to begin by comparison with latent Dirichlet analysis (LDA) [23]. In LDA, the conditional probability of the ith word in the corpus corresponding to document d_i belonging to latent topic z_i is given by [24]:

$$P(z_i = j|\boldsymbol{z}_{-i}, \boldsymbol{w}) \propto \frac{n_{-i,j}^{(w_i)} + \beta}{n_{-i,j}^{(\cdot)} + W\beta} \cdot \frac{n_{-i,j}^{(d_i)} + \alpha}{n_{-i,\cdot}^{(d_i)} + K\alpha} \tag{29}$$

The first ratio is merely the probability of word w_i under topic j, where the second ratio is the probability topic j in document d_i. $n_j^{(w)}$ is the number of times the sampler assigns the word w to topic j, and $n_j^{(d)}$ is the number of times a word from document d is assigned to topic j. $n_{-i}^{(\cdot)}$ is a count which does not include the current assignment of z_i. A dot $(\cdot)$ is used to indicate when one of the three dimensions (i.e., topic, word or document) is not set to a specific value. α and β are both Dirichlet hyper-parameters.

Note that LDA assigns a topic to each word within each document. ICTR, on the other hand, assigns a topic to each positively-rated item at each time step. In other words, when a user positively rates an item we then associate that item to the user in analogy to a word being associated with a document in LDA. Due to their similarities, a similar equation can be written down for ICTR:

$$P(z_i = j|\boldsymbol{z}_{-i}, \boldsymbol{w}) \propto \frac{n_{-i,j}^{(x_i)} + \eta}{n_{-i,j}^{(\cdot)} + T\eta} \cdot \frac{n_{-i,j}^{(u_i)} + \lambda}{n_{-i,\cdot}^{(d_i)} + M\lambda} \tag{30}$$

The first ratio expresses the probability of a positive rating on item x_i under topic j, where the second ratio is the probability of a positive rating in topic j under user i. These correspond respectively to the proportion of times an item x_i is assigned to topic j and the proportion of times a topic j is assigned to a user u_i. T is the total number of recommendations made, and M the total number of users. η and λ are the Dirichlet hyper-parameters from which we sample arm-dependencies ϕ_k and user preferences $\boldsymbol{p}_m$, respectively:

$$\boldsymbol{p}_m|\lambda \sim \mathrm{Dir}(\lambda) \tag{31}$$

$$\phi_k|\eta \sim \mathrm{Dir}(\eta) \tag{32}$$

Where $p_m \in \mathbb{R}^K$ and $\phi \in \mathbb{R}^{K\times N}$. From the vector of arm-dependencies and user preferences we sample items $x_{m,t}$ and latent topics $z_{m,t}$, respectively.

$$z_{m,t}|p_m \sim \mathrm{Multi}(p_n) \tag{33}$$

$$x_{m,t}|\phi_k, z_{m,t} \sim \mathrm{Multi}(\phi_k) \tag{34}$$

Without loss of generality we assume $k = z_{m,t}$, which is the mechanism by which user preferences are associated to item clusters. Note that a user u_i with no prior positive ratings $n_{-i,j}^{(u_i)} = 0$ has an equal chance of being assigned to any topical arm-cluster z_i, although they do not have equal probability of being assigned an item x_i, since within any given cluster certain items will be favored over others.

We assume without loss of generality that the sampled arm $x_{m,t}$ is equivalent to arm n selected by user m at time t (i.e., $x_{m,t} = n$). Then, we can rewrite Equation 30 in the following form:

$$P(z_{m,n} = k|\boldsymbol{z}_{-i}, \boldsymbol{x}) \propto \frac{\eta'_{kn}}{\sum_{n=1}^{N} \eta'_{kn}} \cdot \frac{\lambda'_{mk}}{\sum_{k=1}^{K} \lambda'_{mk}} \tag{35}$$

Where,

$$\eta' = n_{-i,j}^{(x_i)} + \eta \text{ and } \lambda' = n_{-i,j}^{(u_i)} + \lambda \tag{36}$$

From a purely mathematical perspective, we can evaluate the expectation over a Dirichlet distribution $\theta|\alpha \sim \mathrm{Dir}(\alpha)$, and $\theta, \alpha \in \mathbb{R}^K$:

$$E[\theta_k] = \frac{\Gamma(\sum_{i=1}^{K}\alpha_i)}{\prod_{i=1}^{K}\Gamma(\alpha_i)}\int_{\theta_k}\theta_k\prod_{i=1}^{K}\theta_i^{\alpha_i-1}d\theta_k \tag{37}$$

If we define:

$$\alpha_i' = \begin{cases} \alpha_i - 1 & i = k \\ \alpha_i & \text{otherwise} \end{cases} \tag{38}$$

Then,

$$\theta_k\prod_{i=1}^{K}\theta_i^{\alpha_i-1} = \prod_{i=1}^{K}\theta_i^{\alpha_i'-1} \tag{39}$$

$$\frac{\sum_{i=1}^{K}\alpha_i}{\alpha_k}\frac{\Gamma(\sum_{i=1}^{K}\alpha_i)}{\prod_{i=1}^{K}\Gamma(\alpha_i)} = \frac{\Gamma(\sum_{i=1}^{K}\alpha_i')}{\prod_{i=1}^{K}\Gamma(\alpha_i')} \tag{40}$$

Where in the last equation we used the property of the gamma function $x\Gamma(x) = \Gamma(x+1)$. Applying Equation 39 and Equation 40 to Equation 37, we have finally

$$E[\theta_k] = \frac{\alpha_k}{\sum_{i=1}^{K}\alpha_i} \tag{41}$$

Therefore, we can think of the components of a Dirichlet hyper-parameter as representing pseudo-counts, since normalizing over them gives the expectation for θ, which is a vector of counts. Finally, using Equation 41 we can rewrite Equation 35 as

$$P(z_{m,n} = k|\boldsymbol{z}_{-i},\boldsymbol{x}) \propto \mathbb{E}[\eta_{kn}'] \cdot \mathbb{E}[\lambda_{mk}'] \tag{42}$$

3.2.2 Probabilistic Matrix Factorization

Matrix factorization (MF) posits that low-dimensional latent user and item vectors, $p_m, q_n \in \mathbb{R}^K$, where typically $K << M, N$, can be learned such that a rating prediction $r_{m,n}$ between user m and item n can be estimated by

$$r_{m,n} = p_m^T q_n \tag{43}$$

MF models can be trained using singular value decomposition (SVD), alternating least squares (ALS) [13] or non-negative matrix factorization (NMF). Probabilistic matrix factorization assumes that the generative distributions underlying user and item latent vectors are spherical multivariate normal, corresponding to observation noise:

$$p_m | \sigma_m, \Sigma_m \sim \mathcal{N}(0, \sigma_m^2 I) \tag{44}$$

$$q_n | \sigma_n, \Sigma_n \sim \mathcal{N}(0, \sigma_n^2 I) \tag{45}$$

where I is the identity matrix. It further posits a rating distribution given by

$$r_{m,n} | p_m, q_n = \mathcal{N}(p_m^T u_n, \sigma^2) \tag{46}$$

Many models have been proposed which combine PMF and topic regression ([22], [21], [25]). ICTR [19] combines PMF and topic regression by sampling latent user vectors p_m according to the details from the previous section, and latent item vectors by

$$\sigma_n | \alpha, \beta \sim \mathrm{IG}(\alpha, \beta) \tag{47}$$

$$q_n | \mu_n, \sigma_n, \Sigma_n \sim \mathcal{N}(\mu_n, \sigma_n \Sigma_n) \tag{48}$$

$\alpha, \beta, \mu_n, \Sigma_n$ are assumed to be fixed hyper-parameters. σ_n is sampled from an inverse-gamma distribution with parameters α and β. Given a rating $r_{m,n}$ we can factorize the posterior of q_n as follows:

$$
\begin{aligned}
&\Pr(q_n|r_{m,t}, p_m, \sigma_n^2, \mu_n, \Sigma_n) \\
&\propto \mathcal{N}(r_{m,t}|p_m^T q_m, \sigma_n^2) \cdot \mathcal{N}(q_n|\mu_n, \sigma_n^2 \Sigma_n) \\
&\propto \exp\left[-\frac{1}{2\sigma_n^2}\left((r_{m,t} - p_m^T q_n)^2 + (q_n - \mu_n)^T \Sigma_n^{-1} (q_n - \mu_n)\right)\right] \\
&\propto \exp\left[-\frac{1}{2\sigma_n^2}\left(-2\left(r_{m,t} p_m^T + \mu_n^T \Sigma_n^{-1}\right) q_n + q_n^T \left(p_n p_n^T + \Sigma_n^{-1}\right) q_n\right)\right]
\end{aligned}
\tag{49}
$$

The argument of the exponential is quadratic and therefore the posterior is normally distributed. We can complete the square by referencing the form of the posterior:

$$
\begin{aligned}
&\mathcal{N}(q_n|\mu_n', \sigma_n'^2 \Sigma_n') \\
&= \exp\left[-\frac{1}{2\sigma_n^2}\left(\mu_n^T \Sigma_n'^{-1} \mu_n - 2\mu_n'^T \Sigma_n'^{-1} q_n + q_n^T \Sigma_n'^{-1} q_n\right)\right]
\end{aligned}
\tag{50}
$$

Comparing Equation 49 with Equation 50 we see that

$$
\Sigma_n' = (p_m p_m^T + \Sigma_n^{-1})^{-1} \tag{51}
$$

$$
\mu_n' = \Sigma_n'(r_{m,t} p_m + \Sigma_n^{-1} \mu_n) \tag{52}
$$

Furthermore, we can factorize the posterior of σ_n^2 according to

$$
\begin{aligned}
&\Pr(\sigma_2^n|r_{m,t}, p_m, q_n, \alpha, \beta) \\
&\int_{q_n} \Pr(\sigma_2^n|r_{m,t}, p_m, q_n, \alpha, \beta) dq_n \\
&\propto \mathrm{IG}(\sigma_n^2|\alpha, \beta) \mathcal{N}(r_{m,t}|p_m^T q_m, \sigma_n^2) \mathcal{N}(q_n|\mu_n, \sigma_n^2 \Sigma_n) \\
&\propto \left(\frac{1}{\sigma_n^2}\right)^{(\alpha + \frac{1}{2}) + 1} \exp\left[-\frac{\beta}{\sigma_n^2}\right. \\
&\left. -\frac{1}{2\sigma_n^2}\left(r_{m,t}^2 \mu_n^T \Sigma_n'^{-1} \mu_n - 2\mu_n'^T \Sigma_n'^{-1} q_n + q_n^T \Sigma_n'^{-1} q_n\right)\right]
\end{aligned}
\tag{53}
$$

Then,

$$\begin{aligned}\alpha' &= \alpha + \frac{1}{2} \\ \beta' &= \beta + \frac{1}{2}(\mu_n^T \Sigma_n^{-1} \mu_n + r_{m,t}^2 - \mu_n'^T \Sigma_n'^{-1} \mu_n')\end{aligned} \tag{54}$$

3.2.3 Particle Filtering

The presentation in this section is based on [26] and [27]. Particle Filtering (PF) relies on importance sampling (IS) to numerically approximate the expectation under a distribution $p(x)$ for which we cannot draw samples directly (but can be evaluated anywhere) by sampling from another distribution $q(x)$, called the *importance distribution* with corresponding *importance weights* given by:

$$w^{(l)} \propto \frac{\tilde{p}(x^{(l)})}{\tilde{q}(x^{(l)})} \tag{55}$$

Where $\tilde{p}(x)$ and $\tilde{q}(x)$ are the unnormalized distributions of $p(x)$ and $q(x)$, respectively.

Particle filters are a subclass of sequential Monte Carlo (SMC) methods, which, unlike the Kalman or extended Kalman filters, are applicable to non-linear-Gaussian emission densities. Particle filters are used to approximate the posterior at time $t+1$ by drawing a set of samples $\{z_{t+1}^{(l)}\}_{l=1}^{L}$ from the posterior $\Pr(z_t|x_t)$ along with a set of weights $w_t^{(l)}$, otherwise known as the the particle's “fitness" at time t. Then,

$$\Pr(z_{t+1}|x_t) = \sum_l w_t^{(l)} \Pr(z_{t+1}|z_t^{(l)}) \tag{56}$$

Where, the weights are given by the normalized likelihood (probability of observables x_t given the state z_t) at time t:

$$w_t^{(l)} = \frac{\Pr(x_t^{(l)}|z_t)}{\sum_l \Pr(x_t^{(l)}|z_t)} \tag{57}$$

In the case of Rao-Blackwellised Particle Filtering (RBPF) [28] we again constrain ourselves to importance distributions with the Markovian property with importance weights given by:

$$w_t \propto \frac{p(y_t|y_{1:t-1}, r_{0:t})p(r_t|r_{t-1})}{q(r_t|y_{1:t}, r_{1:t-1})} \tag{58}$$

The simplest choice, albeit not the most efficient, is to choose $q(r_t|y_{1:t}, r_{1:t-1} = p(r_t|r_{t-1})$ so that the $p(r_t|r_{t-1})$ cancels with the denominator in Equation 58.

3.2.4 Offline Evaluation Methods

One difficulty which arises in bandit problems but not in the supervised learning setting is the inherent data incompleteness problem. In bandit settings we have only the historical log of user-item ratings from which to assess the performance of our algorithms, but our data is incomplete in the sense that we only observe feedback for the items that were rated. To complicate matters further, our observations are likely to be biased by the recommendation engine currently in production. We can imagine a maximally biased historical log consisting of only ratings for recommended items in the one extreme, and in the other, a purely unbiased historical log consisting of wholly user-driven, instinctive interactions, free of influence from the legacy recommendation engine. We can also picture a second bias-free scenario, in which the recommended items in historical log were sampled randomly from a uniform distribution.

Now, if we proceed under the concession of bias, we can nevertheless relax the assumption of a uniformly randomized item generator to that of any randomized item generator. However, the lessening of constraints on our assumptions and the reduction of bias is not free, as it incurs the expense of increasing estimation variance via decreased data efficiency, since the replayer method is a form of rejection sampling which trades bias for variance [29]. The idea of the replayer is the run the algorithm sequentially through the historical log of ratings, and if at any given time-step the item recommended by the new system matches that found in the historical log we update the score (typically click-through-rate (CTR) or accuracy) and model parameters accordingly, but otherwise do nothing and proceed to the next time-step. Because rejection sampling results in decreased data efficiency, we are in practice sometimes forced to restrict the set of items under evaluation to the top N most popular so as to mitigate the severity of the overall rejection rate.

In the absence of sufficient data to perform a replayer estimation, we can turn instead to leave-on-out methods, where for each user we holdout for training all available user histories except for the most recent example [30]. To mitigate the amount of time consumed ranking all items, [31] samples 100 items outside of the users support set (history of rated items) and ranks them. The metrics which are typically assumed in this setting to evaluate the performance of the ranked list are the Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG). HR simply checks whether the test item is inside the top-N of the ranked list.

3.3 Problem Definition Setting

3.3.1 Collaborative Topic Regression

Collaborative topic regression (CTR) combines topic modelling and CF [22] using a PMF framework, where the rating r_{ij} for user i on item j is assumed to come from a normal distribution with mean given by the inner product of latent user and item feature vectors:

$$r \sim \mathcal{N}(u_i^T v_j, c_{ij}^{-1}) \tag{59}$$

where u_i, $v_j \in \mathbb{R}^d$ are the user and item latent feature vectors, respectively. The precision parameter c_{ij} gives the confidence for rating r_{ij} and is defined heuristically as:

$$c_{ij} = \begin{cases} a & r_{ij} = 1 \\ b & r_{ij} = 0 \end{cases} \tag{60}$$

with $a > b > 0$, since we are more confident in clicks indicating positive sentiment and less confident in the absence of a click indicating negative feedback. The sampling procedure is as follows:

$$\epsilon_j \sim \mathcal{N}(0, \lambda_v^{-1} I_k) \tag{61}$$

$$\theta_j \sim \text{Dirichlet}(\alpha) \tag{62}$$

$$\tag{63}$$

Letting $v_i = \theta_j + \epsilon_j$.

$$\begin{aligned} u_i &\sim \mathcal{N}(0, \lambda_u^{-1} I_K) \\ v_j &\sim \mathcal{N}(\theta_j, \lambda_v^{-1} I_k) \end{aligned} \tag{64}$$

where θ_j is the topic component for the article. Furthermore,

$$\mathbb{E}[r_{ij} | u_i^T, \theta_j, \epsilon_j] = u_i^T (\theta_j + \epsilon_j) \tag{65}$$

Similar to LDA, documents are considered to be multivariate distributions of topics z_{jn}, and topics multivariate distributions of words w_{jn}. For every document j we associate a distribution of topics θ_j, and for every topic we associate a distribution of words described by the simplex $\beta_{z_{jn}}$:

$$z_{jn} \sim \text{Mult}(\theta_j) \tag{66}$$

$$w_{jn} \sim \text{Mult}(\beta_{z_{jn}}) \tag{67}$$

Lastly, we sample from Equation 59 to generate the reward associated with user i and item j.

3.4 Multi-View Interactive Collaborative Topic Regression

Traditional collaborative filtering (CF) algorithms such as alternating least squares (ALS) [13], non-negative matrix factorization (NMF), neural collaborative filtering (NCF) [31] or Bayesian personalized ranking (BPR) [32] have no mechanism for including contextual information (contextual variables are defined as covariates which describe users and items individually or simultaneously) and are unable to make out-of-sample (i.e., cold-start) predictions. Contextual bandits (CB) [33] and factorization machines (FM) [34] can learn on combined contextual and rating data and make out-of-matrix predictions by modifying the input data accordingly. However, when there are large numbers of distinct users or items, or when there are categorical variables, documents composed of words, or feature ontologies, the dimensionality of the design matrix can blow up. CB scales poorly with dimensionality, and while FM scales linearly, both models suffer from the curse of dimensionality. Furthermore, these algorithms cannot be combined with multi-armed bandit (MAB) policies such as Thompson Sampling (TS) or Upper Confidence Bound (UCB).

There are several state-of-the-art models which address both the curse of dimensionality and the cold-start problem simultaneously. Probabilistic matrix factorization (PMF) [20] reduces the dimension of the design matrix and is similar to traditional matrix factorization (MF). Collaborative topic regression (CTR) [22], combines PMF [20] with latent topic modeling, and learns on item-specific (but not user-specific) contextual variables.

Interactive collaborative filtering [17] combines PMF with MAB policies such as epsilon-greedy, Thompson Sampling (TS), upper confidence bound (UCB) and GLM-UCB. The probabilistic frameworks mentioned above address the cold-start problem by relying on pre-specified priors. However, using diffuse priors, as these models do, results in poor predictive performance on cold-start events. We therefore propose a model which personalizes priors using user-item feature dependencies to improve cold-start recommendations.

Interactive collaborative topic regression (ICTR) [19] is another probabilistic algorithm which explicitly models item dependencies as user preference clusters. The built-in modeling of arm dependencies helps the algorithm learn faster, but otherwise it too depends on diffuse priors for generating cold-start recommendations. The model we propose is similar, but instead of modeling arm dependencies it models user-item feature dependencies. Taking inspiration from BayesMatch (BM) [18], a multi-view probabilistic clustering algorithm, we develop RatingMatch (RM), which clusters positively associated (i.e., via implicit or explicit ratings) user and item features. When combined with PMF and a bandit policy we call the resulting algorithm *multi-view interactive collaborative topic regression* (MV-ICTR).

Note that our framework, inspired from CTR [22] does allow for the integration of both the ICTR from [19] with RatingMatch to leverage the strengths of both models. Note that CTR does not consider rating data in the procedure for learning its topic components, relying on latent Dirichlet analysis (LDA) [23], whereas BayesMatch is formulated to explicitly leverage ratings as the associative bodies (i.e., user and item feature sets grouped in proportion to rating as opposed to words grouped by merit of being contained within the same document).

MV-ICTR has dimensionality reduction built-in for improved performance and reduced computational complexity, separating the tasks of cold-start recommendation and online personalization and thereby improving expected short and long-term user experiences.

The core idea idea behind CTR [22] is to estimate reward in a manner similar to PMF

$r_{ij} = u_i^T v$, but with item vector $v_j = \theta_j + \epsilon_j$ represented as the sum of a topic component θ_j and a PMF offset component ϵ_j. Arguably, we want want to express the user vector in a similar manner with $u_j = \theta_i^{(u)} + \epsilon_i^{(u)}$. Then,

$$r_{ij} = (\theta_i^{(u)} + \epsilon_i^{(u)})^T (\theta_j^{(v)} + \epsilon_j^{(v)}) \tag{68}$$

We assume initially (i.e., before any ratings) that $\mathbb{E}[\theta_i^{(u)}] = \mathbb{E}[\theta_j^{(v)}] = 0$ and that $\theta_i^{(u)} \perp\!\!\!\perp \epsilon_j^{(v)}$ and $\theta_j^{(v)} \perp\!\!\!\perp \epsilon_i^{(u)}$, where $\perp\!\!\!\perp$ denotes statistical independence, then

$$\mathbb{E}[r_{ij}] \approx \mathbb{E}[\theta_i^{(u)T} \theta_j^{(v)}] \tag{69}$$

And herein lies the fundamental problem with the introduction of a user topic component $\theta_i^{(u)}$, it must be learned in congruence with $\theta_j^{(v)}$ such that Equation 69 is sufficiently accurate for cold-start users and items. This is precisely the problem we address in this chapter. If we are successful, we will have obtained a built-in method for addressing the cold-start problem in a way that treats users and items symmetrically.

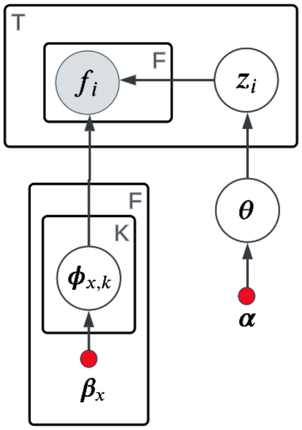


Figure 6. The RatingMatch (RM) component of MV-ICTR, probabilistic diagram for generating topic components.

When user-item implicit or explicit rating pairs exist we are afforded the opportunity to associate user-item covariates. Specifically, user-specific features $f_i^{(u)}$ and item-specific features $f_j^{(v)}$ for any observed user-item pair (i, j) can be concatenated:

$$f_{ij} = [f_i^{(u)}; f_j^{(v)}] \in \mathcal{R}^F \tag{70}$$

where $F = F^{(u)} + F^{(v)} \in \mathcal{R}$ is the combined number of user-specific and item-specific features. MV-ICTR follows a similar line of thinking as [22] with ratings given by:

$$r_{ij} \sim \mathcal{N}(u_i^T v_j, \sigma^2) \tag{71}$$

Now, we modify Equation 64 to

$$
\begin{aligned}
u_i &\sim \mathcal{N}(\chi_i^{(u)}, \lambda_u^{-1} I_K) \\
v_j &\sim \mathcal{N}(\chi_j^{(v)}, \lambda_v^{-1} I_k)
\end{aligned} \tag{72}
$$

giving offset components to both user and items, where both are drawn from spherical normal distributions:

$$
\begin{aligned}
\epsilon_i^{(u)} &\sim \mathcal{N}(0, \lambda_u^{-1} I_K) \\
\epsilon_i^{(v)} &\sim \mathcal{N}(0, \lambda_v^{-1} I_K)
\end{aligned} \tag{73}
$$

we have finally latent user and item feature vectors, respectively:

$$
\begin{aligned}
u_i &= \chi_i^{(u)} + \epsilon_i^{(u)} \\
v_i &= \chi_j^{(v)} + \epsilon_j^{(v)}
\end{aligned} \tag{74}
$$

Then, the conditional expectation of reward r_{ij} is given by

$$
\mathbb{E}[r_{ij} | \chi_i^{(u)}, \chi_j^{(v)}, \epsilon_i^{(u)}, \epsilon_j^{(v)}] = (\chi_i^{(u)} + \epsilon_i^{(u)})^T (\chi_j^{(v)} + \epsilon_j^{(v)}) \tag{75}
$$

where $\chi_i^{(u)}$ and $\chi_j^{(v)}$ give the conditional probabilities of user and item cluster assignments, respectively, and $\epsilon_i^{(u)}$ and $\epsilon_j^{(v)}$ their corresponding PMF offsets. Note that Equation 75 is identical in structure to Equation 68, which is what was desired. The vectors $\chi_i^{(u)}$ and $\chi_j^{(v)}$ are learned via the proposed RatingMatch (RM) procedure depicted in Figure 6. The sampling procedure for RM is given below:

1. Draw the global distribution over clusters $\theta_i \sim \text{Dirichlet}(\alpha)$

2. For each cluster $(k = 1, ..., K)$ and for each feature $(x = 1, ..., F)$

   (a) Draw $\phi_{x,k} \sim \text{Dirichlet}(\beta_x) \in \mathcal{R}^{V_x}$

3. For each user-item pair (i, j)

    (a) Draw cluster assignment $z_i \sim \text{Mult}(\theta_i)$

    (b) Draw user feature value $v_x \sim \text{Mult}(\phi_{x,z_i})$

To develop a procedure for learning $\chi_i^{(u)}$ and $\chi_j^{(v)}$ we first note that the expectations for Dirichlet variables $\theta_k^{(u)}$ and $\phi_{i,k}^{(u)}$ are given by:

$$\mathbb{E}[\theta_k] = \frac{n_k + \alpha}{\sum_{k=1}^{K}(n_k + \alpha)} \tag{76}$$

$$\mathbb{E}[\phi_{i,k}] = \prod_{x=1}^{F} \frac{n_{kf_{xv}} + \beta_x}{\sum_{v=1}^{V_x}(n_{kf_{xv}} + \beta_x)} \tag{77}$$

where $n_{kf_{xv}}$ is the number of times the feature x belonging to ontology f with value v associated with datapoint i is assigned to the k^{th} cluster. Note that $n_{kf_{xv}}$ can be generalized as the corresponding *sum of ratings points*. With this generalization, we therefore allow ratings $r_{i,j} \in [0, \infty)$.

It is helpful to think of each feature x as having their own matrix of counts of size $K \times V_x$. In general, a feature x can have multiple values $v = [v_1, ..., v_R]$, such as a movie with multiple genres. Then, we take

$$n_{kf_{xv}} = \sum_{r=1}^{R} n_{kf_{xv_r}} \tag{78}$$

Each feature is equipped with its own ontology, such as movie genres, user occupations, or skills associated with jobs. The in or out-of-sample probability of assignment for $z_m^{(u)} = k$ for user m with user-specific covariates $f_m^{(u)}$ is therefore given by:

$$\begin{aligned}\chi_{m,k}^{(u)} &\equiv P(z=k|f_m^{(u)}) \\ &= P(z=k)\prod_{x\in F^{(u)}} P(f_{m,xv}^{(u)}|z=k) \\ &\propto \mathbb{E}[\theta_k]\cdot\mathbb{E}[\phi_{m,k}^{(u)}]\end{aligned} \tag{79}$$

Where,

$$\mathbb{E}[\phi_{m,k}^{(u)}] \equiv \prod_{x\in F^{(u)}} \frac{n_{kf_{xv}^{(u)}}+\beta_x}{\sum_{v=1}^{V_x}(n_{kf_{m,xv}^{(u)}}+\beta_x)} \tag{80}$$

Note that $\mathbb{E}[\phi_{m,k}^{(u)}]$ is defined the same as Equation 76 but with the product being taken over user-associated features only. A similar equation exists for items, with the n^{th} item's topic component given by $\chi_{n,k}^{(v)} \equiv P(z=k|f_n^{(v)})$. Equation 79 tells us that the user or item latent components can be obtained by simply “plugging-in" the associated feature values. For training we use collapsed Gibbs sampler similar to that derived in [18] and [24]. The latter found that Gibbs was faster to convergence than either expectation propagation (EP) or variational inference (VI) in learning latent Dirichlet allocation (LDA) [23] components. The conditional probability of cluster assignment is given below:

$$P(z_i^{(u)}=k|z_{-i}^{(u)}) \propto (n_{k,-i}+\alpha)\prod_{x=1}^{F}\frac{n_{kf_{xv},-i}+\beta_x}{\sum_{v=1}^{V_x}(n_{kf_{xv},-i}+\beta_x)} \tag{81}$$

where $n_{kf_{xv},-i}$ is defined the same as $n_{kf_{xv}}$ but with the i^{th} datapoint removed. A similar equation can be written for items. Therefore, when running the collapsed Gibbs procedure it is okay to train on all non-zero user-item ratings, where ratings $r_{i,j} \in [0,\infty)$.

Furthermore, note that there are numerous ways we could contrive to generate user and item offset parameters $\epsilon_i^{(u)}$ and $\epsilon_j^{(v)}$ other than PMF, for instance, using ICTR [19]. However, proceeding with PMF we can analytically compute the posterior distribution for the user matrix U, where each row of U is a user latent vector u_i. Letting, $\delta_{ij} = \{(i, j) :$ user i rated item $j\}$ we have,

$$\begin{aligned} &P(U|R,V;\sigma^2,\sigma_u^2,\sigma_v^2) \propto P(U)P(R|U,V) \\ &\propto \prod_{i=1}^{M} \mathcal{N}(u_i|\epsilon_i^{(u)},\sigma_u^2) \prod_{j\in\delta_{ij}} \mathcal{N}(r_{ij}|u_i^T v_j,\sigma^2) \end{aligned}$$

Taking the logarithm of the result gives:

$$\begin{aligned} &\sum_{i=1}^{M} \frac{-1}{2\sigma^2} \left[\frac{\sigma^2}{\sigma_u^2}(u_i - \epsilon_i^{(u)})^T (u_i - \epsilon_i^{(u)}) + \sum_{j\in\delta_{ij}} (r_{ij} - u_i^T v_j)^T (r_{ij} - u_i^T v_j) \right] \\ &= \sum_{i=1}^{M} \frac{-1}{2\sigma^2} \left[u_i^T \left(\sum_{j\in\delta_{ij}} v_j v_j^T + \frac{\sigma^2}{\sigma_u^2} I \right) u_i - 2u_i^T \left(\sum_{j\in\delta_{ij}} r_{ij} v_j + \epsilon_i \right) + \sum_{j\in\delta_{ij}} r_{ij}^2 + \epsilon_i^T \epsilon_i \right] \\ &= \sum_{i=1}^{M} \log P(u_i|\mu_i,\Sigma_i) \end{aligned}$$

The posterior, like the prior, is a normal distribution. The update equations are therefore given by:

$$\begin{aligned} u_i &= (D_i^T D_i + \lambda_u I_K)^{-1}(D_i^T r_i + \epsilon_i^{(u)}) \\ v_j &= (B_j^T B_j + \lambda_v I_K)^{-1}(B_j^T r_i + \epsilon_i^{(v)}) \\ \Sigma_i^{(u)} &= (D_i^T D_i + \lambda_u I_K)^{-1}\sigma^2 \\ \Sigma_i^{(u)} &= (D_i^T D_i + \lambda_u I_K)^{-1}\sigma^2 \end{aligned} \tag{82}$$

where

$$\begin{aligned} D_i &= \sum_{\delta_{ij}=1} v_i v_i^T \\ B_i &= \sum_{\delta_{ij}=1} u_i u_i^T \end{aligned} \tag{83}$$

are the feature or design matrices, and $\lambda_u = \sigma^2/\sigma_u^2$ and $\lambda_v = \sigma^2/\sigma_v^2$. These can be viewed as the posterior after observing ratings $R = [r_{ij}]$. The corresponding updates are therefore similar to those given by the usual matrix factorization equations [22], [17] and which also appear in Contextual-Bandits (CB) [33], and also reassemble the coefficient estimation for ridge regression.

Note that in CB the design matrices can be in general high-dimensional, and that inversion operations have complexity $O(n^3)$ where n is the number of dimensions. Algorithms which perform dimensionality-reduction or latent variable modeling therefore have potentially vastly improved computational times.

3.5 Results and Discussion

To test the efficacy of our proposed algorithm we experimented on the Movie Lens 100K dataset, which was the only dataset we were able to locate with both ratings and covariates for all user and items. Regarding movies, we extracted two features ($F^{(v)} = 2$) release decade and genre. There were 8 unique decades ranging from 1920 to 1990 and 19 genres within the genre ontology. Regarding users, we used gender, occupation, and age measured in units of decades and rounded to the nearest integer. All features were treated as categorical. The ratings in MovieLens 100K were explicit and ranged from from 1 to 5. These were mapped to 1 if the rating was greater than or equal to 4 and 0 otherwise. We included only the 100 most popular (i.e., most rated) movies in our analysis, which helped to improve data efficiency in the offline evaluation experiments. After filtering, there remained 29.9K user-item ratings.

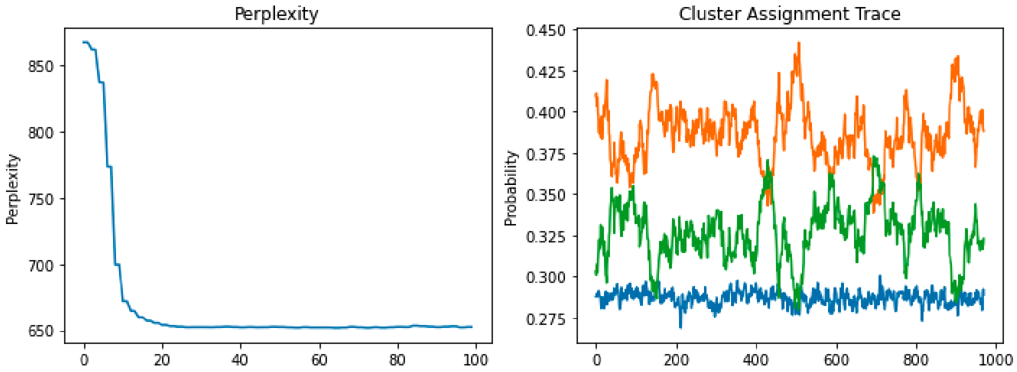


Figure 7. MV-ICTR MovieLens 100K global cluster assignment trace with dimension of latent space given by $d = 3$ over 1000 training dataset iterations, and associated perplexity over 100 training dataset iterations.

Due to the interactive nature of RatingMatch a rejection sampling approach was taken to evaluate the historical log of ratings data [33]. The approach allows us to acquire an unbiased estimate of the model performance but at the expense decreased data efficiency. The replayer method works by stepping through each rating event in the historical log and making a recommendation. If the recommended item does not correspond with the item from the historical log the event is simply discarded (rejected) and the algorithm proceeds to the next event. If the items do correspond, then the model parameters (if applicable) and rating score are updated. We therefore expect roughly 300 impressions (i.e., instances where the recommended item matches that in the historical log).

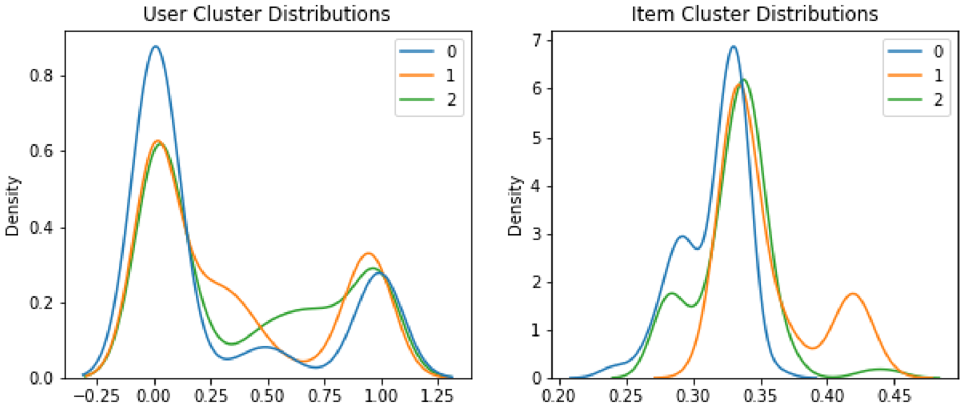


Figure 8. MV-ICTR MovieLens 100K distributions of user and item clusters components $\chi^{(u)}_{n,k}$ and $\chi^{(v)}_{m,k}$, respectively, with the number of latent dimensions $d = 3$, and $k \in [d]$. Taking the average of components over n and m, we observed multi-modal peaks about 0 or 1 for users, and multi-modal peaks about 1/3 for items. The dimorphism in user and item distributions is likely due to the fact that no features are shared between user and items in this dataset.

The data was ordered chronologically according to the event timestamps and split in half for training and testing. 97.7% of datapoints in the test set are cold-start, meaning that either the user or the item rated were not found in the training set. If the online recommender included offset variables we trained these in the training set, and otherwise learned user-item rating-based components “on-the-fly" in the test set. All reinforcement learning algorithms we’re implemented with Thompson Sampling (TS), and in all cases where applicable, we chose the latent dimension of the design matrix or matrices to be $d = 3$. We tested five different algorithms:

Random: Arm recommendations were randomly generated according to a uniform distribution.

Interactive Collaborative Filtering (ICF): ICF was performed with a Thompson Sampling policy.

Collaborative Topic Regression (CTR): Vanilla LDA was trained on the training set and used as the item-offsets in the test set. Following [22] we chose user and item precision parameters $\lambda_u = 0.01$ and $\lambda_v = 100$, respectively. User and item rating-based latent vectors were learned sequentially on the test set, and policy decisions made with TS.

Interactive Collaborative Filtering (ICTR): ICTR was directly applied to the test set with no pre-training on the training set with a TS policy for online inference. Following [19] we chose the dimension of the latent user-item vectors to be $d = 3$, and the number of particles $B = 10$.

MV-ICTR: RM learned user-item feature dependent offset vectors on the training set via collapsed Gibbs sampling. The offsets were then combined to untrained rating-based user and item vectors with $\lambda_u = \lambda_v = 1$ and $\sigma = 0.01$ and implemented with a TS policy on the test set. RM was trained for 1000 iterations over the training set (see Figure 7 and Figure 8).

Due to the stochastic nature of the recommendations and the restrictive size of the test set we trialed each algorithm 10 times and reported the average rating for recommended items over all trials for each algorithm. Perplexity, the typical measure for convergence in language models, was used to measure RM convergence, and is given by

$$\text{Perplexity} = \exp\left(\sum_{i=1}^{T} \log p(f_i)\right) \tag{84}$$

where,

$$\begin{aligned} p(f_i) &= \sum_{z=1}^{d} p(f_i, z) \\ &= \sum_{z=1}^{d} p(f_i | z = k) p(z = k) \end{aligned} \tag{85}$$

where $f_i = [f_n^{(u)}; f_m^{(v)}]$ are the concatenated user and item features for the i^{th} datapoint.

3.5.1 Conclusion

MV-ICTR was found to significantly increase the average rating on the test set by 13.5% points over ICTR (see Figure 9), the second best performing state-of-the-art algorithm tested in the application. We attribute the large jump in performance to the fact that the test set was 97.7% composed of cold-start datapoints.

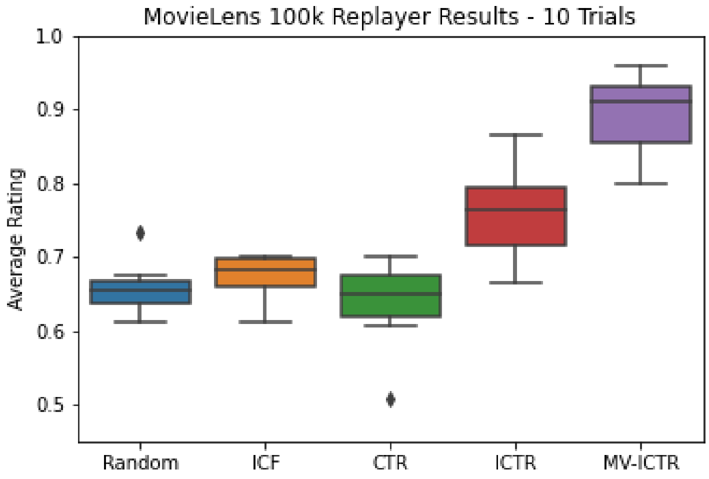


Figure 9. Random, Thompson Sampling (TS), CTR, ICTR and MV-ICTR average rating of recommended items over 10 replayer trials. Rating performed significantly better than ICTR, the second highest performing model, with an average increase in performance of 13.5% (two-sided t-test and $p < 0.01$).

In summary, MV-ICTR combines RatingMatch, a multi-view clustering algorithm, with PMF for interactive recommendation. It saves computational time by reducing the dimension of the design matrix, and further saves on computation by learning the RM topic components offline. It was found to significantly outperform comparable leading state-of-art algorithms in the space of sequential or interactive bandit-based recommender systems, particularly on datasets with high percentages of cold-start users and items. It also generalizes well: RM is a multi-view clustering algorithm, capable of clustering user and items with overlapping, partially overlapping, or non-overlapping feature sets. RM also allows for out-of-sample predictions and for ratings assignments $r_{ij} \in [0, \infty)$, and because it is Bayesian, it is able to easily manage missing data.

MV-ICTR utilizes all of the strengths of RM and generalizes via PMF. Albeit, with enough rating data ICF should perform comparably, in real life applications user interaction data can be sparse and limiting, with high item turnover rates leading to high cold-start percentages. MV-ICTR solves the cold-start and personalization problem simultaneously, making it an optimal choice for applications such as article or job recommendation when user and item contextual information are available.

3.5.2 Future Work

In future work, we would like to experiment with building a fully online topic model where both topic and rating components are updated after receiving feedback. We are also interested in combining RM with ICTR for simultaneous user-item feature and item dependency modelling. Such a model could implement a particle filtering (PF) algorithm for online inference. We would also like to extend the research to different datasets, and to experiment with more bandit algorithms and different parameter values of the latent dimension, additionally comparing performance against computational efficiency.

References

[1] J. D. Angrist and J.-S. Pischke, *Mostly Harmless Econometrics: An Empiricist's Companion*. Princeton University Press, Dec. 2008, ISBN: 0691120358.

[2] R. McElreath, *Statistical Rethinking, A Course in R and Stan*. 2015. [Online]. Available: http://xcelab.net/rmpubs/rethinking/Statistical_Rethinking_sample.pdf.

[3] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. Routledge, 2017.

[4] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[5] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[6] D. Bertsimas, J. Dunn, and N. Mundru, "Optimal prescriptive trees," *INFORMS Journal on Optimization*, vol. 1, no. 2, pp. 164–183, 2019.

[7] M. Amram, J. Dunn, and Y. D. Zhuo, "Optimal policy trees," *Machine Learning*, pp. 1–28, 2022.

[8] H. Laurent and R. L. Rivest, "Constructing optimal binary decision trees is np-complete," *Information processing letters*, vol. 5, no. 1, pp. 15–17, 1976.

[9] M. Biggs, W. Sun, and M. Ettl, "Model distillation for revenue optimization: Interpretable personalized pricing," in *International Conference on Machine Learning*, PMLR, 2021, pp. 946–956.

[10] S. Wager and S. Athey, "Estimation and inference of heterogeneous treatment effects using random forests," *Journal of the American Statistical Association*, vol. 113, no. 523, pp. 1228–1242, 2018.

[11] N. Kallus, "Recursive partitioning for personalization using observational data," in *International conference on machine learning*, PMLR, 2017, pp. 1789–1798.

[12] L. Zou *et al.*, "Neural interactive collaborative filtering," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 749–758.

[13] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, "Large-scale parallel collaborative filtering for the netflix prize," in *International conference on algorithmic applications in management*, Springer, 2008, pp. 337–348.

[14] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in *Proceedings of the 10th international conference on World Wide Web*, 2001, pp. 285–295.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Second. The MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/the-book-2nd.html.

[16] W. Chu, L. Li, L. Reyzin, and R. Schapire, “Contextual bandits with linear payoff functions,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, JMLR Workshop and Conference Proceedings, 2011, pp. 208–214.

[17] X. Zhao, W. Zhang, and J. Wang, “Interactive collaborative filtering,” in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 2013, pp. 1411–1420.

[18] A. Maurya and R. Telang, “Bayesian multi-view models for member-job matching and personalized skill recommendations,” in *2017 IEEE International Conference on Big Data (Big Data)*, IEEE, 2017, pp. 1193–1202.

[19] Q. Wang *et al.*, “Online interactive collaborative filtering using multi-armed bandit with dependent arms,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 8, pp. 1569–1580, 2018.

[20] A. Mnih and R. R. Salakhutdinov, “Probabilistic matrix factorization,” *Advances in neural information processing systems*, vol. 20, 2007.

[21] S. Purushotham, Y. Liu, and C.-C. J. Kuo, “Collaborative topic regression with social matrix factorization for recommendation systems,” *arXiv preprint arXiv:1206.4684*, 2012.

[22] C. Wang and D. M. Blei, “Collaborative topic modeling for recommending scientific articles,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 448–456.

[23] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[24] T. L. Griffiths and M. Steyvers, “Finding scientific topics,” *Proceedings of the National academy of Sciences*, vol. 101, no. suppl_1, pp. 5228–5235, 2004.

[25] K. Wang, W. X. Zhao, H. Peng, and X. Wang, “Bayesian probabilistic multi-topic matrix factorization for rating prediction.,” in *IJCAI*, vol. 16, 2016, pp. 3910–3916.

[26] P. M. Djuric *et al.*, “Particle filtering,” *IEEE signal processing magazine*, vol. 20, no. 5, pp. 19–38, 2003.

[27] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006. [Online]. Available: /bib/bishop/Bishop2006/Pattern-Recognition-and-Machine-Learning-Christophe-M-Bishop.pdf,/bib/bishop/Bishop2006/978-0-387-31073-2_sm.pdf, https://www.microsoft.com/en-us/research/people/cmbishop/#!prml-book.

[28] A. Doucet, N. De Freitas, K. Murphy, and S. Russell, “Rao-blackwellised particle filtering for dynamic bayesian networks,” *arXiv preprint arXiv:1301.3853*, 2013.

[29] L. Li, W. Chu, J. Langford, and X. Wang, “Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms,” in *Proceedings of the fourth ACM international conference on Web search and data mining*, 2011, pp. 297–306.

[30] X. He, T. Chen, M.-Y. Kan, and X. Chen, “Trirank: Review-aware explainable recommendation by modeling aspects,” in *Proceedings of the 24th ACM international on conference on information and knowledge management*, 2015, pp. 1661–1670.

[31] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 173–182.

[32] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, “Bpr: Bayesian personalized ranking from implicit feedback,” *arXiv preprint arXiv:1205.2618*, 2012.

[33] L. Li, W. Chu, J. Langford, and R. E. Schapire, “A contextual-bandit approach to personalized news article recommendation,” in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 661–670.

[34] S. Rendle, “Factorization machines,” in *2010 IEEE International conference on data mining*, IEEE, 2010, pp. 995–1000.