A scalable model for online contextual music recommendations

EMANUELE COVIELLO, Amazon Music
KATHERINE ELLIS, Amazon Music
FABIAN MOERCHEN, Amazon Music

Engaging personalized recommendations are critical to the success of music streaming services. In this paper we experiment with a scalable design for building online personalized contextual recommenders across different music styles. We break down the architecture prominently into an online contextual recommender selection step and an online contextual content selection and ranking step. We discuss the value of this architecture by presenting experimental results on a genre-based personalized recommender on the home page of a global music streaming service.

CCS Concepts: • **Computing methodologies** → **Online learning settings**; *Learning from implicit feedback*; *Ranking*; • **Information systems** → *Content ranking*; **Personalization**.

Additional Key Words and Phrases: Contextual recommendations, Personalized Recommendations, Music Recommendations, Multi Armed Bandits, A/B Testing

ACM Reference Format:
Emanuele Coviello, Katherine Ellis, and Fabian Moerchen. 2021. A scalable model for online contextual music recommendations. 1, 1 (September 2021), 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Beyond making accessible a huge catalog of songs, videos, and podcasts, including original content, streaming services bring value to users worldwide through personalized recommendations. In particular, to keep users engaged, it is key to serve personalized recommendations based on user taste and context and present these recommendations in an intuitive way. People generally relate to their music in terms of a richer list of styles, aesthetics, specific artists, and even moods; preferences for these styles and attributes can differ depending on current activities or intents. We are therefore motivated to provide faceted (i.e., attribute-based) recommendations along a rich set of such dimensions, and experiment with contextual models to select the right facets and content.

In particular, we are interested in a scalable design for personalized recommendations, that could support a portfolio of faceted recommenders that can be served to users. We do not make limiting assumptions on how these recommenders can be served, and we envision a system that is compatible with a multiplicity of apps on mobile devices or smart speakers. For example, such a design might be used to serve personalized recommenders on a home page. Similarly, on more specific pages (such as search pages), it could be adapted to provide personalized recommenders that fit the theme of the page (e.g., the most relevant sub-genres given a genre search to refine the search, or a "<genre> like <artist>"

Authors' addresses: Emanuele CoviellоAmazon Music, emacov@amazon.com; Katherine EllisAmazon Music, kathee@amazon.com; Fabian MoerchenAmazon Music, moerchen@amazon.com.

Manuscript submitted to ACM

widget given an artists search", etc...). Alternatively, when interfacing with a voice-assistant, such a model might help serving requests that includes specific music styles. For example, for a request to "Play rock", it might contribute top candidates from a personalized recommender and from a new releases recommender built around rock or its sub-genres.

In this paper we describe one possible conceptual architecture of a contextual personalization engine and we present experiments on a genre-based personalized recommender on the home page of a music streaming service as a motivating example. In particular, the architecture consists in two steps, first determining a theme within each template to present to a user (for example, a sub-genre for the personalized genre recommender) and then populating the recommender with a personalized selection of music entities that are relevant to the theme and also contextually relevant.

The contribution of this work is that it evaluates a modular workflow for building personalized recommenders, which is split into a contextual recommender/theme selector, and a contextual content selector and re-ranker. The modularity allows to improve and troubleshoot each component separately, and increases interpretability of the systems compared to methods that jointly select theme and content as in [11]. In addition, compared to existing approaches that use contextual MABs for online music recommenders such as [3] and [11], in this work we serve multiple entity types beyond just playlists. Finally, by ensembling candidates from multiple content generation strategies in the final recommendations, we can effectively leverage a variety of content in the same recommender (for example the most popular albums from a genre, together with a personalized selection of playlists and albums with the highest affinity to the customer).

The rest of the paper is organized as follows. We discuss related work in Section 2. We describe the architecture design in Section 3. We presented results in Section 4 and draw conclusions in Section 5.

2 RELATED WORK

This work builds upon a significant amount of related work in using contextual multi-armed bandit (MAB) algorithms to make recommendations. Many recommendation tasks are inherently a case of bandit feedback, because user feedback is biased by which content was shown to the user by the current production recommendation system. Part of a bandit algorithms task is to manage the tradeoff between exploiting its current understanding of the optimal recommendation for a user and exploring areas of the recommendation space that have more uncertainty. Notably, Li et al. propose framing personalized news recommendation as a contextual bandit problem and propose the LinUCB algorithm for learning [9]. Shani et al. define recommendations as a sequential decision making process via a Markov decision process [15]. Other approaches blend collaborative filtering and bandit algorithms using k-nearest neighbors [14] or sparse graph representations [10], or use sub-modular optimization for diversification of rankings [12].

To address the ranking use-case, in which the task is to present multiple recommended items to a user at a time, Radlinski describe an approach for learning diverse rankings [13]. Kveton et al. use cascading bandits to learn rankings [8]. Authors from Amazon use contextual bandits with a position-based model (PBM) to learn rankings for music recommendation [6].

In the context of music recommendation, Wang et al. use multi-arm bandits to serve interactive, personalized music recommendations [16]. Authors from Spotify present a method based on contextual bandits to jointly optimize recommendations and explanations [11] and authors from the music streaming service Deezer describe the use of multi-armed bandits to personalize carousels of music recommendations [3]. Dragone et al. explore the formulation of reward functions in contextual bandits for music recommendation [5]. They key areas of novelty where this work differs from Spotify's published approach [11] is that the authors provide more control on the selection of recommender templates and seeds, and that multiple types of entities beyond playlists are presented by the recommender. Compared

to Deezer's publication [3], this work addresses themed recommenders (instead of a generic playlist recommender) and uses an ensemble model for content generation.

3 ARCHITECTURE

At a high level, generating music recommendations can be comprised by two steps. First, predict the user's intent, meaning determine what, at some level of granularity, they are looking for. Second, select content from the catalog that matches that intent. We present a model that aims to decompose the recommender system architecture into these separate steps so that progress can be made to improve each component in parallel.

We can slice the catalog by a variety of dimensions, or facets, to create different granularities of content groupings. Examples of dimensions are genre, era, similarity to a seed artist, familiarity to the user, etc. We can define facet templates, logic that defines recommenders based on combinations of facet values to serve an individual user intent (e.g., finding music similar to their favorite artist, or discovering new music they're not familiar with). Some facet templates require seeds (e.g., the artist for a "similar to <artist>" template) that additionally need to be selected. On mobile apps, we are not limited to selecting a single intent at a time, but we can instead select a variety of diverse facet templates that maximize the chances of the user finding content to engage with.

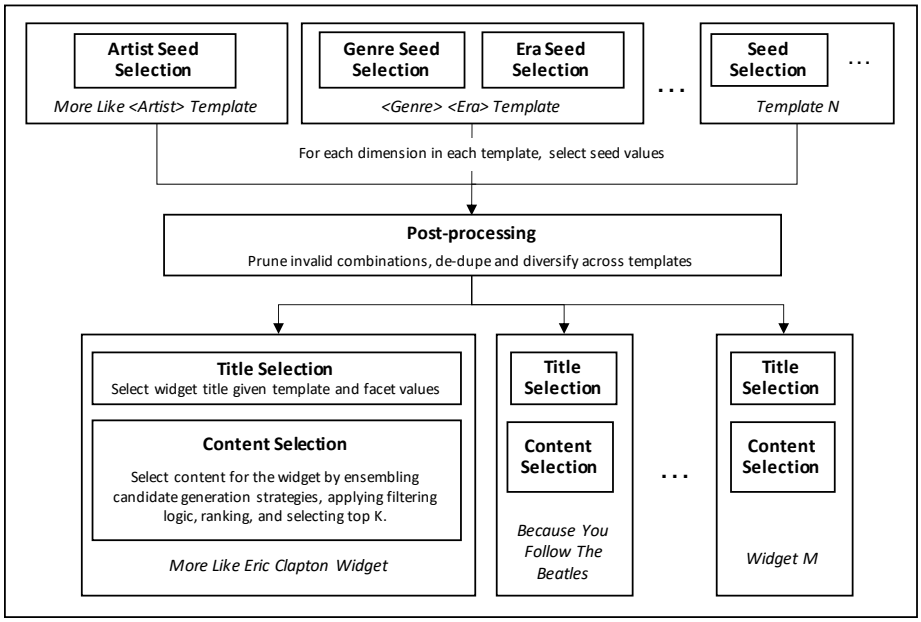
Figure 1 outlines the conceptual vision for an architecture for contextual personalized recommendations, which consists of five main components. **(1) Recommender selection** selects a high-recall set of facet templates to serve user+context request. **(2) Seed selection**, for each facet template (if applicable), selects the K best candidate seeds for each dimension. We experiment with a contextual multi-armed bandit to select seeds; more details are provided in section 3.2 below. **(3) Post-processing**, given the pool of seeded facet templates, de-duplicates and diversifies to select the final set of recommenders to serve a user+context request. **(4) Title selection** selects a title given the seeded facet template. In experiments presented in this paper, this is a simple lookup based on pre-defined facet template titles and seed values (however, in practice it can be optimized [11]. **(5) Content selection** selects the content to serve in each recommender given the seed(s). We experiment with contextual multi-armed ranking bandit to select content; more details are provided in section 3.3 below.

We chose to approach both seed selection and content re-ranking with linear bandits trained with Thomson sampling (linTS) due its simplicity and effectiveness, which comes from balancing exploration with exploitation, adaptability to changes in online traffic, and efficiency in how it handles data since each example is processed once. An extension of this work would consist in improving our bandit modeling framework. In particular, one interesting direction is the exploration of bandit models explicitly designed for optimizing ranking metrics [1], rather than modeling the problem as reward regression, as in the current linTS formulation. Learning-to-rank formulations can improve performance by allowing the model to focus on optimizations that actually change the ranking of actions, rather than over-indexing on minimizing error in reward prediction. Along the same lines, we can addressed the limitation of a linear formulation by moving to higher-capacity models using deep neural networks as the function combining action and context features. We briefly present preliminary experiments with these models in Section 4.

3.1 Bandit Setup

For both seed selection and content selection, we frame decisions about which seed or content to select as a bandit problem. In a bandit framework, there is an agent (i.e., the seed or content selector), which interacts with an environment (i.e., the user) in a sequential nature. At each time t, the agent receives a request for a recommendation, along with a context vector x_t, a vector of features describing the user and current context. The agent also has available a set of K

Fig. 1. High-level architecture for contextual personalized recommenders



actions (i.e., candidate seeds or candidate entities to be recommended) to be selected from. Each action is described by an action vector a_t, a vector of features describing the action and/or the user's affinity toward the action. In the traditional top-1 multi-armed bandit case, the agent selects a single action to present to the user, and in turn receives a reward, r_t for the action. Based on that reward, or feedback, the agent can update its action selection policy. Over time, the goal of the agent is to learn an action selection policy that maximizes the cumulative reward $\sum_{t=1}^{T} r_t$.

We use a linear Thompson Sampling bandit (linTS) [2], which is a Bayesian model that samples actions according to the probability that the action maximizes the expected reward. Given context vector x_t and action vector a_i, the linTS bandit assumes the reward for action i at time t is generated from an (unknown) distribution with mean $f(x_t, a_i)^T\theta$, where $\theta \in \mathbb{R}^d$ is a fixed but unknown parameter. The function f could be a somewhat arbitrary combination of context and action features, from naive concatenation of features to a deep neural network.

In the top-K recommendation case, the agent must select a ranked list of K actions to present to the user, and in turn receives a list of rewards a_{tk} corresponding to each action selected. We model the learning-to-rank problem using the position based model (PBM) [4], which makes the simplifying assumption that we can decompose the relevance of an action in a given context from the position in which the action was displayed. More concretely, the true reward r_{tk} for the action at position k in time t is not directly observed, but instead we observe a masked version of the reward $c_{tk} = o_{tk}r_{tk}$, where o_{tk} is a binary random variable indicating whether the the true reward was observed. In the position bias model, we assume that the observation variable o depends only on the position, not on context or action features, so that $Pr(o_{tk} = 1|x_t, a_{tk}, k) = Pr(o = 1|k)$. The probabilities $Pr(o = 1|k) := p_k$ are called position bias probabilities, and are estimated independently [7, 17]. This allows us to update the linTS bandit to compute the expected payoff of ranking a given action i in position k, conditionally on the action: $\mathbb{E}[c_{tk}|a_i] = p_k f(x_t, a_i)^T\theta$. The posterior distribution of θ can be estimated in closed form for linear Thompson Sampling [6].

3.2 Seed Selection

The goal of the seed selector is to select the best seed(s) for each dimension of a recommender (e.g., the artist in "More like <artist>" or genre in "<Genre> For You") given a user and context. A seed selector can be divided in two components. First, **seed candidate generation** returns a high-recall list of valid seed values for the user. It collects valid values from potentially multiple sources. For example, an artist seed candidate generator could include artists the user has played frequently, artists the user follows, artists the user has recently listened to, etc.

Then, a **seed ranker** selects the best seed value using a contextual multi-armed bandit with features describing the request context and each candidate seed value. We use a top-1 bandit to select a single seed per facet template and user.In the experiments discussed below, we model the reward as a binary feedback signal, with a value of 1 if the user clicked on the widget and a zero otherwise.

In our experiments we have included a variety of context and action features, including the user affinity to the candidate seed value (artist, genre, etc.) based on different windows of time, information about the seed value (e.g., genre taxonomy) and several contextual features, such as type of device, temporal information, a user's affinity toward familiar content or newly released content, etc.

3.3 Content Selection

The goal of content selection is to select the ranked list of content to display for a given facet template and seed value(s). As with seed selection, content selection can be divided into two components. First, a set of **content generation strategies** each return a high-recall list of entities that are eligible to be served in a recommender, given the seed(s). For example, a popularity-based content generation strategy sources the most popular albums tagged with the seed genre, while a personalized content generation strategy sources albums tagged with the seed genre that the user has highest affinity towards (based on a collaborative filtering model trained offline). These strategies can be shared across recommenders where it makes sense; i.e., a popularity-based content generation strategy is useful for a variety of facet templates, with just the filtering logic varying.

After generating a candidate pool of content, a **content ranker** , or **content ensembler** selects the K best entities to serve in a recommender from the pool of candidate entities sourced in the previous step. The ranker is a contextual multi-armed ranking bandit that uses features describing the request context and each candidate entity to rank and select the best entities to serve a given request.

While the bandit modeling framework, feature definitions, and hyperparameters are shared across facet templates for engineering consistency, each facet template learns independent model weights. This is because in contrast to seed selection, the same types of entities are recommended across facet templates (albums, playlists, etc.), so feature definitions do not need to change; but learning independent model weights allows us to learn different semantics in different use-cases. For example, we could learn that popularity is an important feature for predicting the success of an entity in a New Releases widget but less important in a discovery-oriented widget. In our experiments we model the reward as a binary feedback signal, with a value of 1 if the user clicked on the entity and a zero otherwise.

Features used in the content selection model include the user affinity to the entity according to an offline-trained collaborative filtering model, user affinity to the entity type (album, playlist, etc.), information about the entity (e.g., popularity, release date) and several contextual features as described above.

Table 1. Results of genre seed selection experiment.

	Average effect (global)	Cross-territory deviation
Fraction of users with at least one successful playback on the genre recommender		
relative lift (%)	48.07*	10.19
Average number per user of successful playbacks on the genre recommender		
relative lift (%)	73.39*	17.36

* Statistically significant with $p < 0.0001$

4 RESULTS

In this section we describe experimental results based on online traffic on the home page of a music streaming service. In particular, we focused on a genre-based recommender. In Section 4.1 we present an experiment on the seed selection model, and in Section 4.2 results on the content ensembling and ranking model.

In order to single out the effect of our contextual multi-armed bandit models on the genre recommender, we look at specific recommender-level metrics. In particular, we look at the average number per user of successful playback attributed to the genre recommender (average successful playbacks), and the fraction of home page users who have at least one successful playback through the recommender (users with successful playback).

In order to baseline linear Thomson sampling (LinTS) to alternative models for selection and ranking, we performed offline experiments over the data that we logged during the online experiments presented in Sections 4.1 and 4.2.[1] In particular, we compared our model to neural LinTS (NN LinTS), that uses a neural network with a final LinTS layer, trained with offline replay.[2] In both seed selection and content ranking, the NN LinTs model underperformed LinTS, with a 0.36% (not statistically significantly different at 5% significance) and 12.0% relative drop in performance, respectively. In the ranking case, we also compared the ranking LinTS bandit to a neural network trained offline with a ranking loss [1]. The ranking neural network performed with a 2.3% relative improvement over the linTS bandit, but the improvement was not statistically significant at 5% significance. While we plan to investigate deeper architecture further, these experiments validate the modeling decisions we made for the online experiment.

4.1 Seed selection experiments

We ran an online AB test on the genre-based recommender globally for a duration of 14 days, to compare the MAB-based seed selection strategy described in Section 3.2 to a baseline that selects a random seed from the top-20 genres and sub-genres from a user history. Note that the latter strategy is mildly personalized in the sense that it selects genres from a user history, however it does not use context features to optimize the selection. Note that the setup of this experiment did not include the content ensembling presented in Section 3.3, and so given the genre seed we simply retrieved the top Albums and Playlists from the genre and rank them by user affinity based on a collaborative filtering model.

The increase in widget-level engagement was substantial. In Table 1 we report the lift in average successful playbacks and users with successful playback on the genre recommender. The cross-territory deviation reports standard deviation of the lift across 10 territories worldwide. We see that the contextual MAB model drove a 48.07% relative increase in

[1]We split the data in two temporal partitions. We use the first partition to warm start a model, and the second to simulate online data that the model would see in production. We report performance based on offline evaluation using inverse propensity score (IPS) weighting.
[2]The hidden layers are updated with stochastic gradient descent.

the fraction of users who had at least a successful playback on the genre recommender during the 14-days window of the experiment, and a relative increase in successful playback on the recommender of 73.39%. We noticed that the decision consistency [3] of the model grew over time, reaching 70% after the second day, and stabilizing to 89.35% overall, suggesting that the MAB has reached a high decision consistency (low exploration).

Finally, since music genres can be organized into a taxonomy, we looked at how often the MAB model selects genres at different taxonomy levels. In particular, in Table 2 we group statistics for taxonomy level 1 (i.e., top-genres), level 2 (i.e., sub-genres) and level 3 or above (i.e., micro-genres). We see that the MAB selects genres at both levels, favoring level 1 genres about 74% of the time, and level 2 or level 3+ genres 13% and 12% of the time respectively. Since it seems that level 3+ micro-genres are receiving higher rewards in average, one might wonder why the MAB would not select these genres more often. There are several possible explanations, including that the MAB model is successfully balancing multiple signals, and selecting micro-genres in the contexts they are most likely to perform well in.

Table 2. Statistics for seed genres by taxonomy level

Genre Taxonomy Level	Proportion	Avg Reward relative to Level 1
Level 1	74.4%	1.000x
Level 2	13.0%	0.870x
Level 3+	12.6%	1.113x

4.2 Content ensembling experiments

We ran an online AB test on the genre recommender globally for 14 days to evaluate the content ensembler described in Section 3.3.

In the control group, the existing candidate generator used popularity-based search (i.e., query the for the most popular (non-personalized) albums and playlists matching the given genre seed) and ranked candidates using a heuristic based on a single personalized feature derived from a collaborative filtering model. We added a new candidate generator based on personalized search (i.e., query a personalization service for the albums/playlist with highest affinity to the user filtered to match the given genre seed), and we tested three methods to ensemble candidates from the two strategies: a ranking based the same user-to-entity affinity feature (T1) as the control ranking, a randomized round-robin interleaving of the candidates from the two strategies (T2), and the ensembler based on the content ranking contextual MAB (T3) described in Section 3.3. Note that the only strategy that uses contextual information for ensembling and ranking is T3.

In Table 3 we report the lift in percent of users with successful playback on the genre recommender and the standard deviation of the lift across 10 territories worldwide. It is clear that ensembling multiple content strategies using a MAB (T3) has a substantial impact on widget-level engagment. Specifically, the MAB-based ensembler performed the best overall, driving a 20.02% relative increase in the fraction of users with successful playback from the genre recommender in a 14-days window (of the experiment). In addition it drove a relative increase in successful playbacks on the recommender of 23.27% (we do not report the metric in the table).

Interestingly, we see that ensembling multiple strategies using heuristics alone is not sufficient, and that using the contextual MAB is key to achieving success. In particular, ranking by user affinity alone (T1) underperforms the baseline,

[3] We define the decision consistency as the fraction of times the chosen action is the action with the max propensity; the propensity is the probability that an action is scored highest by a random weight vector sampled from the posterior, estimated with a Monte Carlo approximation.

Table 3. Results of content selection experiment

Fraction of users with at least one successful playback on the genre recommender	Average effect (global)	Cross-territory deviation
T3 relative lift (%)	20.02*	11.63
T2 relative lift (%)	2.46*	1.29
T1 relative lift (%)	-4.32*	2.72

* Statistically significant with p < 0.0001

with a -4.32% relative drop in the fraction of users with a successful playback from the genre recommender (and a drop of -11.55% in average successful playbacks per user). Ranking by user affinity in practice positions all personalized candidates before the popular candidates. These results suggest that personalized candidates are important, but using only user affinity as a ranking signal is not sufficient — music listeners also respond to popularity, recency, etc. and including both richer contextual signals and action features is key to achieving an engaging ranking. Similarly, we see that replacing the MAB with the round-robin heuristic (T2) results in just a mild increase of 2.46% in the fraction of users with successful playbacks.

5 CONCLUSIONS AND FUTURE WORK

In this paper we described models based on contextual MABs for the seed selection and content selection of a personalized genre-based recommender, and we presented experimental results based on online AB tests run globally on the home page of a music streaming service. The experimental results demonstrate that the use of contextual MABs substantially increase user engagement with these recommenders.

REFERENCES

[1] Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. A general framework for counterfactual learning-to-rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 5–14, 2019.

[2] Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*, pages 127–135. PMLR, 2013.

[3] Walid Bendada, Guillaume Salha, and Théo Bontempelli. Carousel personalization in music streaming apps with contextual bandits. In *Fourteenth ACM Conference on Recommender Systems*, pages 420–425, 2020.

[4] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. Click models for web search. *Synthesis lectures on information concepts, retrieval, and services*, 7(3):1–115, 2015.

[5] Paolo Dragone, Rishabh Mehrotra, and Mounia Lalmas. Deriving user-and content-specific rewards for contextual bandits. In *The World Wide Web Conference*, pages 2680–2686, 2019.

[6] Beyza Ermis, Patrick Ernst, Yannik Stein, and Giovanni Zappella. Learning to rank in the position based model with bandit feedback. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2405–2412, 2020.

[7] Zhichong Fang, Aman Agarwal, and Thorsten Joachims. Intervention harvesting for context-dependent examination-bias estimation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 825–834, 2019.

[8] Branislav Kveton, Csaba Szepesvari, Zheng Wen, and Azin Ashkan. Cascading bandits: Learning to rank in the cascade model. In *International Conference on Machine Learning*, pages 767–776. PMLR, 2015.

[9] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670, 2010.

[10] Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 539–548, 2016.

[11] James McInerney, Benjamin Lacker, Samantha Hansen, Karl Higley, Hugues Bouchard, Alois Gruson, and Rishabh Mehrotra. Explore, exploit, and explain: personalizing explainable recommendations with bandits. In *Proceedings of the 12th ACM conference on recommender systems*, pages 31–39, 2018.

[12] Houssam Nassif, Kemal Oral Cansizlar, Mitchell Goodman, and SVN Vishwanathan. Diversifying music recommendations. *arXiv preprint arXiv:1810.01482*, 2018.

[13] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on Machine learning*, pages 784–791, 2008.

[14] Javier Sanz-Cruzado, Pablo Castells, and Esther López. A simple multi-armed nearest-neighbor bandit for interactive recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 358–362, 2019.

[15] Guy Shani, David Heckerman, Ronen I Brafman, and Craig Boutilier. An mdp-based recommender system. *Journal of Machine Learning Research*, 6 (9), 2005.

[16] Xinxi Wang, Yi Wang, David Hsu, and Ye Wang. Exploration in interactive personalized music recommendation: a reinforcement learning approach. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 11(1):1–22, 2014.

[17] Xuanhui Wang, Nadav Golbandi, Michael Bendersky, Donald Metzler, and Marc Najork. Position bias estimation for unbiased learning to rank in personal search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 610–618, 2018.