Content-Based Music Recommendation Using Non-Stationary Bayesian Reinforcement Learning

Article *in* International Journal of Social Ecology and Sustainable Development · January 2022

DOI: 10.4018/IJSESD.292053

Brijgopal Bharadwaj
VIT University

Marimuthu Karuppiah
SRM Institute of Science and Technology

International Journal of Social Ecology and Sustainable Development
Volume 13 • Issue 9

Content-Based Music Recommendation Using Non-Stationary Bayesian Reinforcement Learning

Brijgopal Bharadwaj, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India

Ramani Selvanambi, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, India

Marimuthu Karuppiah, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Delhi-NCR Campus, India*

https://orcid.org/0000-0001-7379-2641

Ramesh Chandra Poonia, Department of Computer Science, CHRIST University (Deemed), India

ABSTRACT

This paper presents a music recommendation system for the offline libraries of songs that employs the concepts of reinforcement learning to obtain satisfactory recommendations based on the various content-based parameters. In order to obtain insights about the effectiveness of the generated recommendations, parallel instances of single-play multi-arm bandit algorithms are maintained. In conjunction to this, the concepts of Bayesian learning are considered to model the user preferences by assuming the environment's reward generating process to be non-stationary and stochastic. The system is designed to be simple, easy to implement, and on-par with user satisfaction within the bounds of the input data capabilities.

KEYWORDS

Audio Features, Beta Sampling, Clustering, Explore-Exploit Dilemma, Independent Bandit Algorithm, Playlist Generation, Recommendation System, Thompson Sampling

INTRODUCTION

The recommendation systems are one of the key features of all the major platforms with business models that rely on the sustenance and quality of user interaction to function smoothly. In the recent years, they are being rapidly incorporated in most of the large-scale products, and are a subject of active research, due to the ever-increasing demand to understand what are the preferences of the user, and how do they evolve.

One of the key challenges that are encountered by the music recommendation systems is the fact that the user is subjected to prefer different type of content at different points of time, based upon his mood, external influences, evolving preferences, content-availability, etc. Thus, a recommendation system (RS) is subjected to a very sparse and stochastic rewarding process. In order to still remain relevant, a RS must be able to adapt accordingly. Thus, it must be the aim of the RS to minimize the abandonment, which occurs when none of the recommended items are relevant to the user.

Another challenge that is faced by almost all the RSs is the well-known "Explore-Exploit" dilemma. It deals with the choice to either explore the available decision space, or to exploit the

DOI: 10.4018/IJSESD.292053 *Corresponding Author

accumulated knowledge and recommend the best possible result. To solve this problem, the traditionally used algorithms include a certain class of them, often referred to as the "bandit" algorithms. They propose various methods of establishing a delicate balance between the choice of exploration and exploitation. The Bayesian approach, commonly referred to as Thompson sampling is one such algorithm that is shown to have optimal regret bounds. However, it is only designed to be used in a single-play and stationary environment. In this paper, we use the Discounted Thompson Sampling (DTS) algorithm to account for the non-stationary environment. This is paired up with the Independent Bandit Algorithm (IBA) to allow the incorporation of the multiple-play scenario.

One of the major concerns with the recommendation systems is the cold-start problem, i.e. when the accumulated data is not sufficient to provide satisfactory results. This is another challenge that needs to be overcome, in order to obtain optimal performance. To address this issue, we incorporate the use of content-based features to establish a base-level similarity measure among the discovered songs in the library, to provide better-than-random results in the initial phase of the recommendation cycle.

LITERATURE SURVEY

The currently available literature on music recommendation systems can be broadly classified into three categories *(X. Wang et. al., 2014):*

- Collaborative Filtering
- Content-based Filtering
- Context-based Filtering

Collaborative filtering (CF) methods are widely used in recommender systems. They provide recommendations based on ratings that users give to items. The results of these techniques are quite good; however, the difficulty in obtaining explicit feedback in the form of ratings from the users causes the sparsity problem, which takes place when the number of available ratings for the items to be recommended is small. CF is powerless when confronted with the new-song problem — it cannot recommend songs without prior usage history. *(D. S. Moreno et. al., 2016, Y. Koren et. al., 2009).*

In the field of Content-based filtering, the main focus is on a systematically detailed analysis of the input audio-signals, and based on the values of a number of computed parameters that help to quantify the various dimensions of the signal,*(Liang et. al., 2015).(V. Murthy et. al., 2018)*provide an informative summarization of the various content-based music information retrieval (CB-MIR) literature works, and helps us to determine the course of action, by observing a birds-eye view of the systems, and learning from them.*(Zhang et. al., 2019)*provided us with the latest innovations that have been published in the field of recommendation systems, using the paradigms of deep learning frameworks.*(X. Wang et al., 2014)* present a RL-based solution to learn user preferences, using a Bayesian model that accounts for both audio content and the novelty of recommendations. The presented single unified model can be used for both music recommendation and playlist generation. They use a generated dataset with explicitly compiled user-ratings per song to train the model, in contrast to the implicit feedback approach adopted in this paper.

(Raj et. al., 2017) provide an algorithm that extends upon the ideas presented by *(William R. Thompson, 1933)*to enable the usage of Thompson sampling approach in a non-stationary environment. *(Radlinski et. al., 2008)* proposes the ranked bandit algorithm, and *(Kohli et. al., 2013)* proposed the independent bandit algorithm, to incorporate bandit algorithms in a multiple-play context. *(Louedec et. al., 2015)* compare these two methods, and propose an efficient implementation of the *Exp3.M* algorithm proposed by *(Uchiya et. al., 2010).*

PROBLEM FORMALIZATION

In order to clearly explain the mathematical intricacies involved, in this section we briefly describe the notations used henceforth. We consider that the RS is exposed to a set of K items, represented as I_i where $i \in \{1, 2, \ldots, K\}$. At each time instance t, m items are selected from the given set and recommended to the user. This set of items is referred to as A_t. When the user interacts with the system, RS receives a reward in accordance to a set of rules R_r. At a given time t, a user is represented by X_t.Z_t is the reward obtained by the system for the set A_t with the relevance vector X_t, and is defined by (1).

$$Z_t = \max_{i \in A_t} X_{i,t}\,. \tag{1}$$

The expectation of Z_t, $E[Z_t]$ is defined as the fraction of users for whom at least one item is relevant in A_t. Maximization of $E[Z_t]$ is logically equivalent to minimization of the abandonment. Using this, we define an optimal set A^* using (2) to be a set of items that leads to at least one click for a majority of users.

$$A^* = \underset{A \in P_m^K}{\operatorname{argmax}} E\left[Z\right] \tag{2}$$

The RS is designed to minimize the difference between the sum of rewards that are obtained by using the algorithm, and the sum of rewards that would have been received if A^* was used. This difference is referred to as the expected regret, $R(T)$, and is represented using (3).

$$R\left(T\right) = T \times EZ^* - \sum_{t=1}^{T} E\left[Z_t\right] \tag{3}$$

The task of finding A^* is an NP-hard problem *(Radlinski et. al., 2008)*. Thus, an easier-to-find suboptimal set is what we aim for. In order to make that possible, *(Kohli et. al., 2013)*proposed the following algorithm, referred to as the Independent Bandit Algorithm (IBA).IBA maintains m parallel instances of single-play multi-armed bandit algorithms (SPBAs), that are aimed at solving the explore-exploit dilemma. In our implementation, we have adopted a modified version of the Bayesian approach, more commonly known as Thompson Sampling, to be used as the SPBA.

The Thompson Sampling (TS) algorithm provides a systematic method to account for the exploration-exploitation trade-off, by performing actions that aim to maximize the expected reward with respect to a randomly drawn belief. In its most widely used implementation, it works by maintaining a prior probability distribution over success for each Bernoulli arm, and sampling from this distribution to select the arm to play. Beta distribution, the conjugate-prior to the Bernoulli distribution, is selected to be the prior. It is characterised by two shape-parameters: α andβ. For each iteration, these parameters are updated in accordance with the received reward.

The Discounted Thompson Sampling (DTS) proposed by *(Raj et. al., 2017)* is an algorithm based upon the traditional TS architecture that aims to provide a mechanism to account for the restless bandit-scenarios, where the reward-generating process is not assumed to be stationary.

This is done by introducing systematic variance in the prior distributions maintained for the unexplored arms, while keeping the expected value almost constant. For this we use a discount factor γ that induces exponential filtering to gradually reduce the influence of the past observations.

Algorithm 1. Independent Bandit Algorithm (IBA)

1:	SPBA_i : single-play bandit algorithm for i^{th} recommendation.
2:	**for** $t = 1, 2, \ldots, T$ **do:**
3:	**for** $i = 1, 2, \ldots, m$ **do:**
4:	$a_i \leftarrow$ *selectItem*(SPBA_i, $K \backslash A_{t,i-1}$)
5:	$A_{t,i} \leftarrow A_{t,i-1} \cup a_i$
6:	**end**
7:	Display A_t to the user, receive X_t
8:	**for** $i = 1, 2, \ldots, m$ **do:**
9:	*Feedback:* $z_i = \begin{cases} 1, & \text{if item } a_i \text{ was used} \\ 0, & \text{otherwise} \end{cases}$
10:	*update*(SPBA_i, z_i)
11:	**end**
12:	**end**

FEATURE EXTRACTION

In order to build the foundations of the considered task of constructing a robust music recommendation system, the literature was systematically reviewed, and the most effective parameters and properties were identified, that could be computed from the input song samples, using the *librosa* library in python. Henceforth, the song-sample from Fig. 1 will be used for the purpose of demonstration.

A. Tempo and Duration:

It has been proposed that the listeners often have a range of common attributes across the songs most played by them. Some of these attributes are the basic musical attributes like Tempo, song-structure, etc. Thus, the input song sample's tempo and duration are considered to be the features for clustering, to account for these observations.

B. Spectral Features:

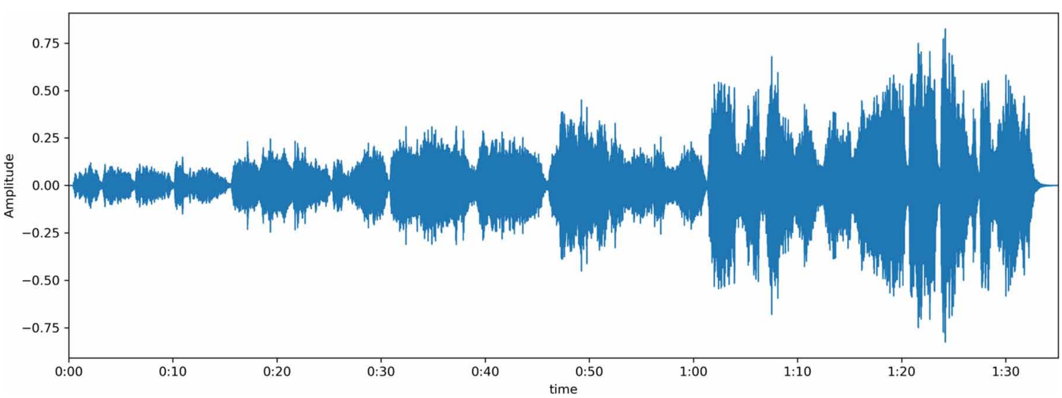
The root mean square (RMS) value of the input sample provides the idea of the energy that each frame of the song is associated with. The temporal distribution of the observed RMS values of the frames, calculated using (4) as seen in Fig. 2,can be a considered a coarse indicator of the loudness of the song in general.

$$RMS\left(m\right) = \sqrt{\frac{1}{N}\sum_{n=-N/2}^{N/2}\left(x\left(n+mh\right)\right)^{2} w\left(n\right)} \quad (4)$$

Algorithm 2. Discounted Thompson Sampling (dTS)

1:	**Parameters:** $\gamma \in (0,1]$, $K \geq 2$, α_0, $\beta_0 \in \mathbb{R}_{\geq 0}$
2:	**Initialization:** $S_k = F_k = 0 \ \forall\, k \in \{1, 2, \ldots, K\}$
3:	**For** $t = 1, 2, \ldots, T$ **do:**
4:	**For** $k = 1, 2, \ldots, K$ **do:**
5:	$\theta_k(t) \sim Beta(S_k+\alpha_0, F_k+\beta_0)$
6:	**end**
7:	Play arm $I_t^\pi = \underset{k}{\operatorname{argmax}}\, \theta_k\left(t\right)$, observe reward $\tilde{r}_t$
8:	Perform a Bernoulli trial: success probability $\tilde{r}_t$
9:	Observe the output r_t
10:	Update: $S_{I_t^\pi} \leftarrow \gamma S_{I_t^\pi} + r_t$
11:	Update: $F_{I_t^\pi} \leftarrow \gamma F_{I_t^\pi} + \left(1 - r_t\right)$
12:	Update: $S_{I_t^\pi} \leftarrow \gamma S_{I_t^\pi}$ & $F_{I_t^\pi} \leftarrow \gamma F_{I_t^\pi}$; $\forall k \neq I_t^\pi$
13:	**end**

Figure 1. Wave-plot of the song sample



The zero-crossing rate (ZCR) is a weighted measure of how many times the signal changed its polarity in a given frame. Empirically it can be observed that ZCR values, calculated using (5), are high for a noisy/percussive signal, and vice versa. The framewise ZCR-values for the input signal are shown in Fig. 3.

$$ZCR\left(m\right) = \frac{1}{2N}\sum_{-N/2}^{N/2} V \tag{5}$$

$$V = sgn\left(x\left(n + mh\right)\right) - sgn\left(x\left(+mh - 1\right)\right) \tag{6}$$

The spectral-centroids (SC) are a characterizing feature of digital signal processing, that elucidate the signal's amplitude distribution (7). As can be observed in Fig. 4, they can also be thought of as the locations of the frame's spectrum's "center-of-mass". The SC values can be associated with the sound's brightness.

$$SC\left(m\right) = \frac{\sum_k f_k X\left(m,k\right)}{\sum_k X\left(m,k\right)} \tag{7}$$

Spectral rolloff (SR) values indicate the frequency values, such that a given percentage of the total spectral energy of the corresponding frame is concentrated below them (Fig. 5). The spectral bandwidth (SB) values are one of the major features used to characterize the power spectrum of the input signal in the discipline of digital signal processing. Their higher order moments can be used to quantify the level of peakedness and asymmetry of the spectrum for a given frame. Thus, the means and variance of their distribution (shown in Fig. 6) are considered in the feature set of the algorithm proposed in this paper.

C. Chroma Features:

The chroma features and chromagrams bear a close relation with the twelve primary scales of music theory, and thus provide various valuable insights in the structure of the melodic and harmonic aspects of a symphony *(Ellis D., 2007)*. They also robustly capture the changes observed in the timbral and instrumentation progressions of the input signal, thus allowing us to perform multi-faceted analysis of the input piece of music.

Chroma features also enable us to have a rough estimation of the musical notes used in the song (Fig. 7), thus further providing useful features to perform clustering analysis. In order to incorporate these features in the process, we include the mean and variances of each pitch class of the chromagram as a feature for each input signal.

D. Mel-Frequency Features:

The Mel-frequency scale is constructed by applying certain non-linear transformations on the conventional frequency scale, in such a way that the frequencies that are equidistant from each other on the Mel-scale are also perceived to be equidistant by the human ear, which is not the case with the former scale. The Mel-Frequency spectrogram of the input sound is shown in Fig. 8.

The spectral envelope thus generated can be coarsely characterized by the Mel-Frequency Cepstrum Coefficients (MFCCs). In order to capture the temporal variations of MFCCs, the mean and variances of their gradients are also considered as some of the features. The Fig. 9 shows the temporal progression of the MFCCs for the given input song.

E. Tonnetz features:

The tonal-centroid features (Tonnetz) are considered to be quite effective at detecting changes in the music harmonies of the input audio signal. They achieve this capability by constructing a conceptual 6-D lattice structure as a representative of the tonal space of music theory, by mapping the 12-bin chroma vectors to the latter's interior space *(Harte et. al., 2006)*. The algorithm is shown to successfully detect shifts in the chord boundaries and harmonies. The tonnetz features for the considered song sample are shown in Fig. 10.

ALGORITHM IMPLEMENTATION

Once the feature vectors for each song in the considered dataset were computed, it resulted in a 1008×503 feature matrix, which was then used to perform the following sequence of operations:

A. Initial Clustering:

One of the major hurdles in front of any recommender system that relies on the concepts of reinforcement learning to operate is the cold-start problem. During the initial phase of operation, the recommender system often gives sub-optimal and barely coherent and consistent results, which could potentially become a major cause of a bad user experience. In order to prevent this incoherency, we perform a clustering operation on the supplied dataset, to obtain some information regarding the songs that show similar characteristics.

The number of clusters (K) to be formed is a hyper-parameter, whose value can be tweaked as seen fit. For the sake of implementation, its value was selected according to (8), and the clustering was performed by transforming the input vectors to a 150-dimensional space, using the Kernel Principle

Figure 2. RMS-values of the song sample

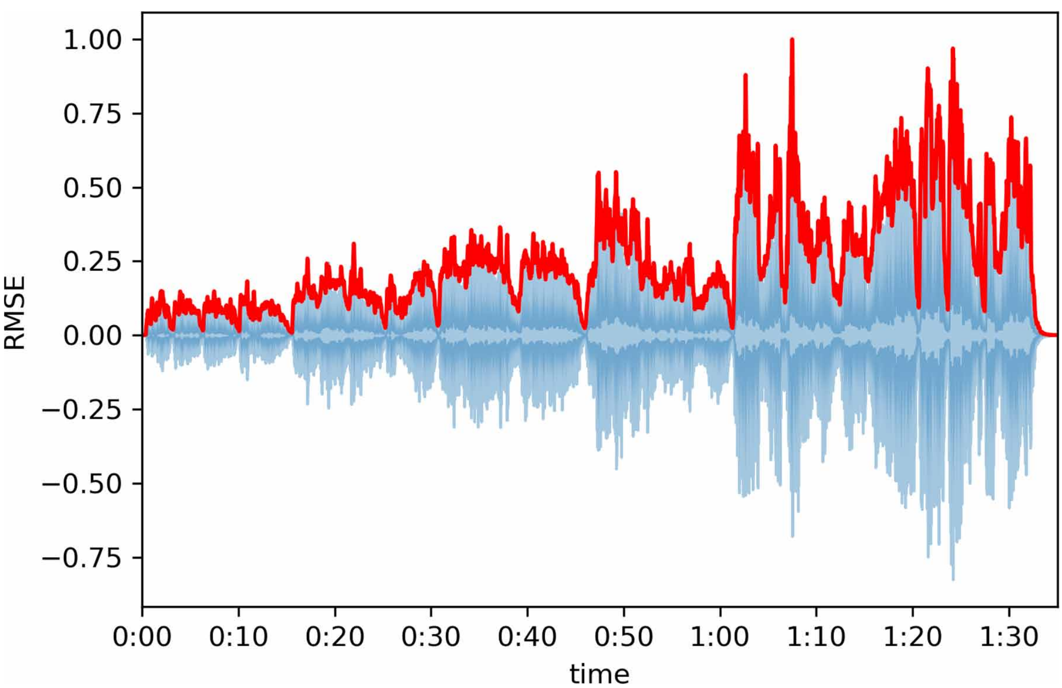


Figure 3. ZCR-values of the song sample

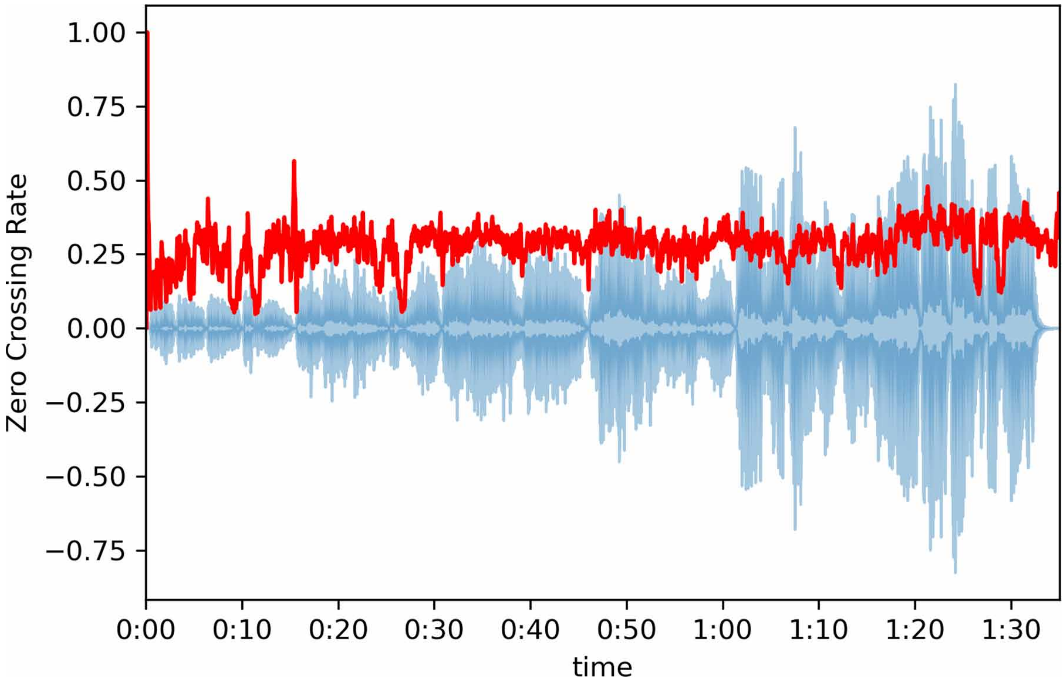


Figure 4. Spectral centroid values for each frame of the song sample

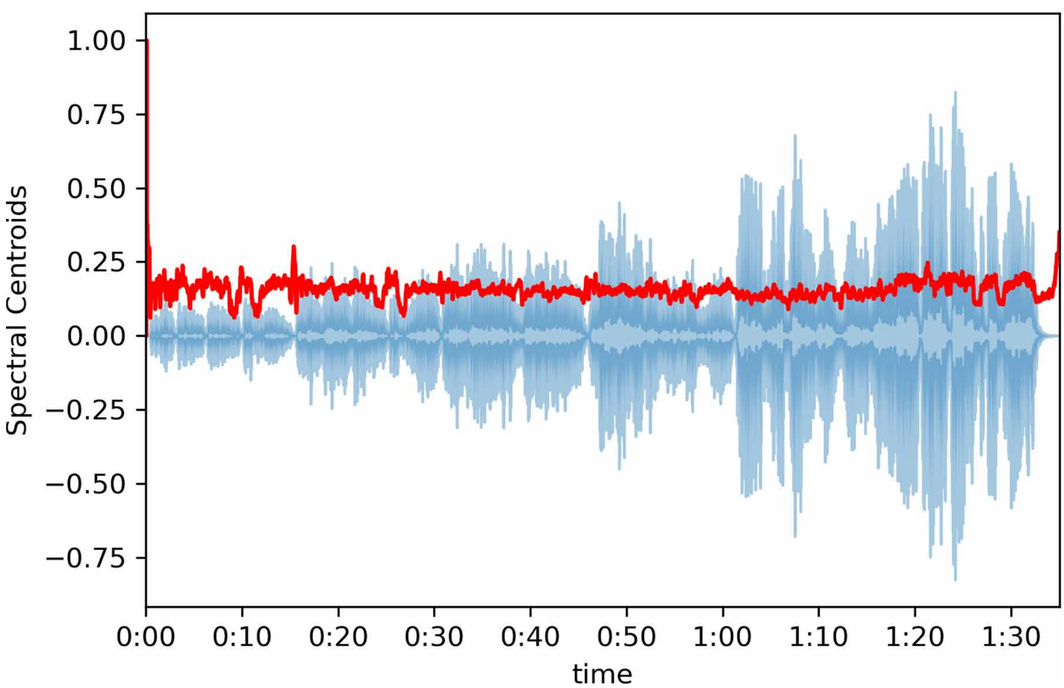


Figure 5. Spectral-rolloff values for each frame of the song sample

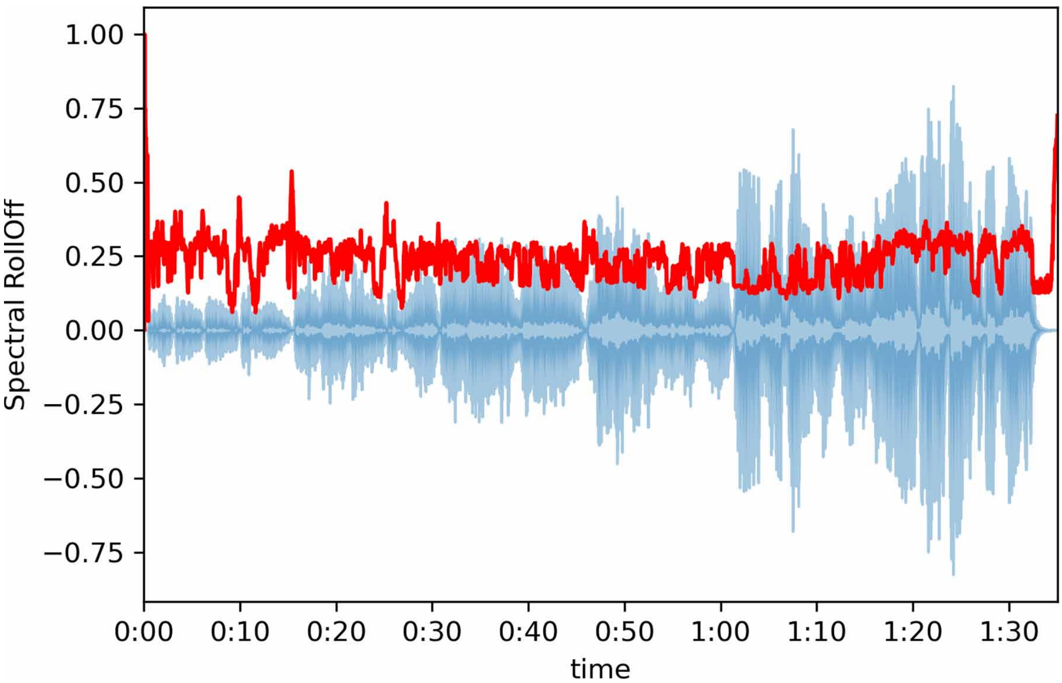


Figure 6. Spectral-bandwidth values for each frame of the song sample

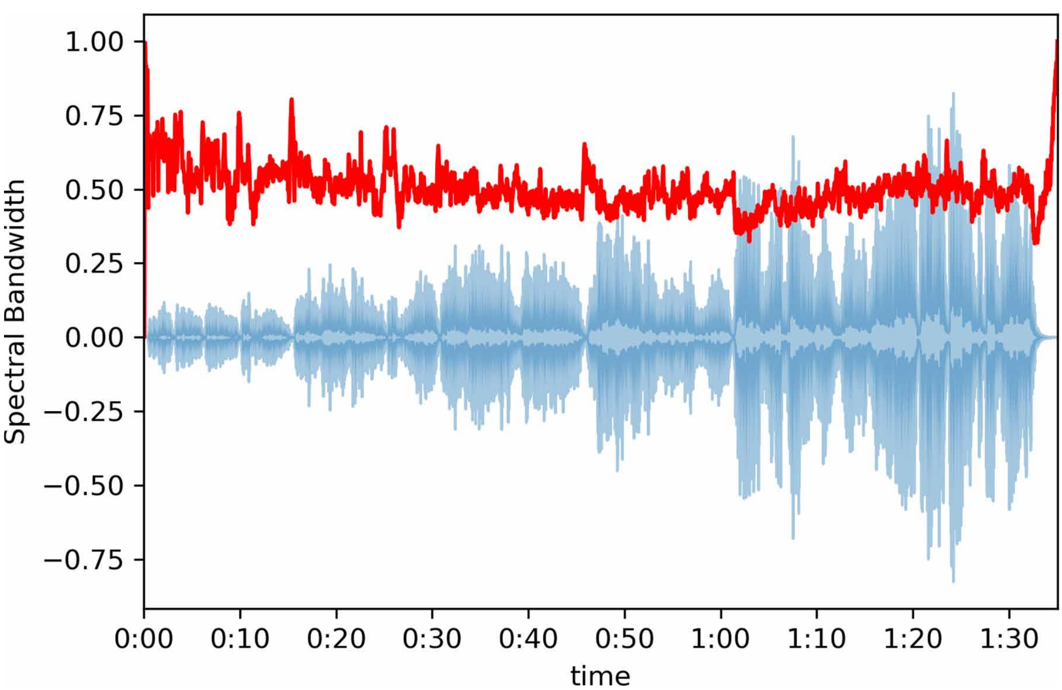


Figure 7. Chromagram representing the 12 musical pitch classes via the chroma-features of the input wave sample

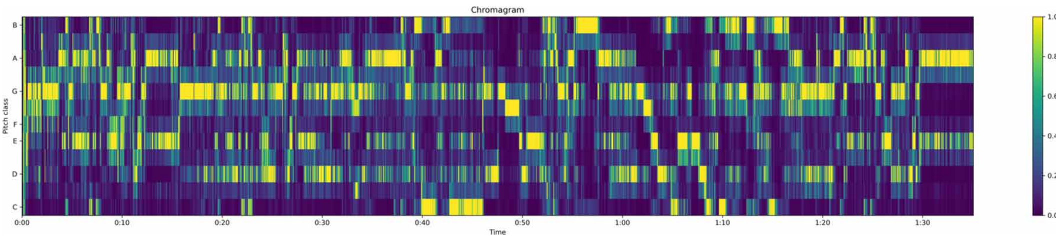


Figure 8. Mel-frequency spectogram of the input wave sample

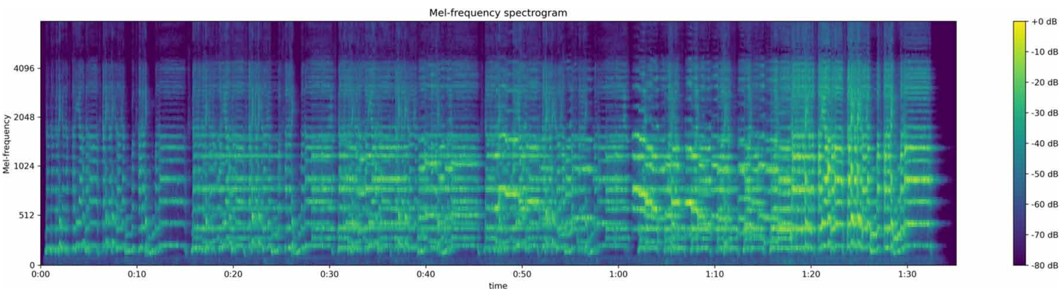
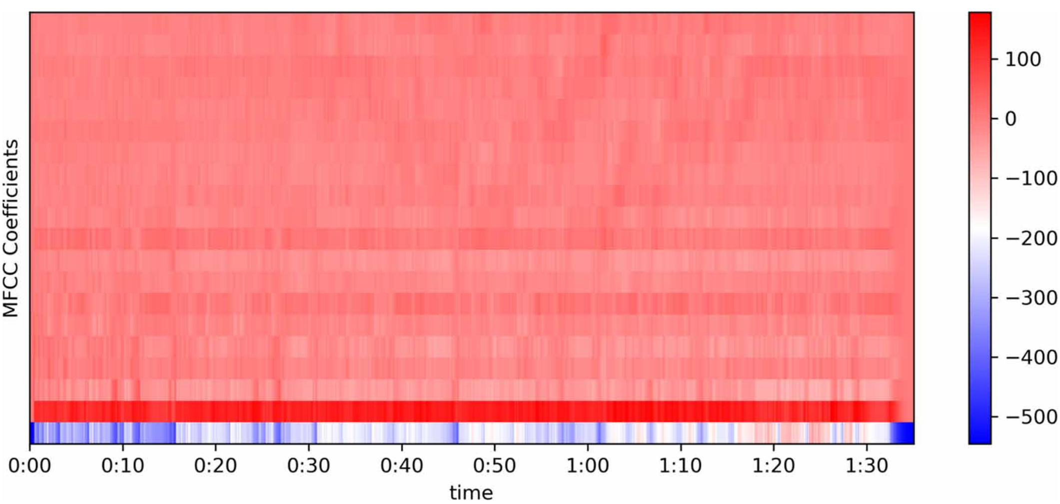


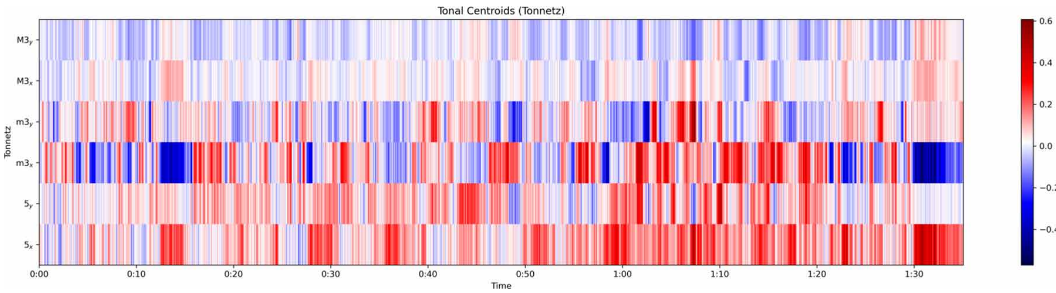
Figure 9. MFCC values for each frame f the song sample



Component Analysis (KPCA) with a *'poly'* kernel for the process of dimensionality reduction. The obtained clusters of songs were then processed according to the following operations, to generate recommendations.

$$K = min\left[2ceil\left(\log_2 N\right),\ mod\left(N,5\right)\right] \tag{8}$$

Figure 10. Tonnetz features for each frame of the song sample



B. Music Recommendation:

This section elaborates on the application of the discussed algorithms to work with a user interface that offers two services: "Recommended Playlists" and "Shuffle". Each section has its own set of parameters that work in coordination to present better results to the user. The following sections discuss each one of these in detail:

1. Recommended Playlists:
   a. We select the number of playlists N_p going to be generated using (9). For each playlist P_i, we define the parameters α_{ij} and β_{ij}, where $i \in \{1, 2, \ldots, N_p\}$, $j \in \{1, 2, \ldots, K\}$. These are the shape parameters of the beta-distributions B_{ij}, and are initially set to be 1. This essentially converts each B_{ij} into a uniform distribution, signifying that for all $x \in [0, 1]$, the probability of obtaining x upon random sampling, $B_{ij}(x)$, is equal. As a result, the expected mean of each distribution B_{ij} is 0.5.

$$No.of\ Playlists\left(N_p\right) = min\left[\frac{K}{2}, 10\right] \tag{9}$$

   b. Each playlist is composed by randomly selecting 20 songs from the corresponding cluster. For each playlist P_i, we randomly sample each distribution B_{ij}, and select the one with the highest sampled value. This becomes the value selected by $SPBA_i$ in Algorithm 1.
   c. We define that a time instant t lasts until the user closes the application. Thus, in this time-period, he/she may select anywhere from none to all the shown recommendations. For each playlist, a user is considered to be engaged, if he/she spends at least three listening minutes (i.e. length of songs heard) on the playlist. The 3-minute criterion is another hyper-parameter, which can be tuned according to the need.
   d. For each playlist P_i, a payoff $\tilde{r}_t$ is issued, according to (10), which lie in [0, 1]. This value is then used to perform a Bernoulli trial, with the success probability as $\tilde{r}_t$.

$$r = \frac{listening\, duration\left(L.D.\right)}{total\, duration\, of\ playlist\left(T.D.\right)} \tag{10}$$

$$reward, r_t = \begin{cases} 0, & if\, L.D. < 5\, min \\ r, & otherwise \end{cases} \quad (11)$$

e. The obtained result of Bernoulli trial is in {0, 1}, and is passed-on as the reward r_t received by the selected playlist by $SPBA_i$, and it is recorded in X_{it}. Based upon the reward, the values of α_{ij} and β_{ij} are updated in accordance to the algorithm 1.

2. Shuffle Feature:
   a. When the user selects a given song to play, the system identifies the cluster it belongs to.
   b. The identified cluster is then screened for finding the nearest neighbours of the song.
   c. The songs are selected in a batch of 20, and an intra-batch randomization operation is performed to obtain the list of songs to be played next. The outputs from all the batches are concatenated to the playing queue.
      e. If the song selected initially is selected from within a suggested or user-created playlist, the songs taken into consideration are only scoped to the playlist.

RESULTS

A. Clustering Analysis:

The dataset of the song-features was subjected to a min-max scaling operation, to scale all the features to the range of 0 to 1.Following that, a kernel-PCA transformation was applied to the feature-set, and 150 vectors were considered, which accounted for about 95% of the variance. These vectors were taken as the input for the clustering operation, which segregated the input songs into 20 separate clusters, such that all the songs within a cluster shared some common characteristics.

B. Recommendation Analysis:

The clusters thus identified were passed-on to a GUI-based application, implemented using React, Redux, and Electron.js. The home page of the application can be seen in Fig. 11. The learning aspects of the application were performed using a vanilla JavaScript implementation, and were accessed via a store-based Redux architecture.

The user is presented with multiple playlists generated using the clustering results. The system maintains a parameter-table to log the current values of all the α_{ij} and β_{ij} corresponding to each playlist and slot. This table is updated in accordance with the previously described algorithm, each time the user interacts with the system.

After carrying-out multiple usability analyses on the system, it was observed that as the application was used for listening songs, over-time it was able to tailor the recommendations using the previously described algorithms, to get a good approximation of the user preferences.Also, two of the playlists generated by the system are presented in Table 1. To quantitatively observe the system performance and the trends it follows, the approach presented by *(X. Wang et al., 2014)* was chosen. Twosister implementations of the same application were developed, only difference being the learning strategy being employed by each implementation.

The first sister implementation employed a random recommendation strategy to decide which items to recommend to the user. It represents a purely explorative strategy for song-recommendation. The other sister implementation employed a greedy recommendation strategy to decide the future recommendations, by recommending the songs that are most interacted-with by the user. This represents a purely exploitative strategy for music-recommendation. These two implementations provide us with the comparison benchmarks to analyze the performance of the presented algorithm.

Figure 11. GUI of the developed application

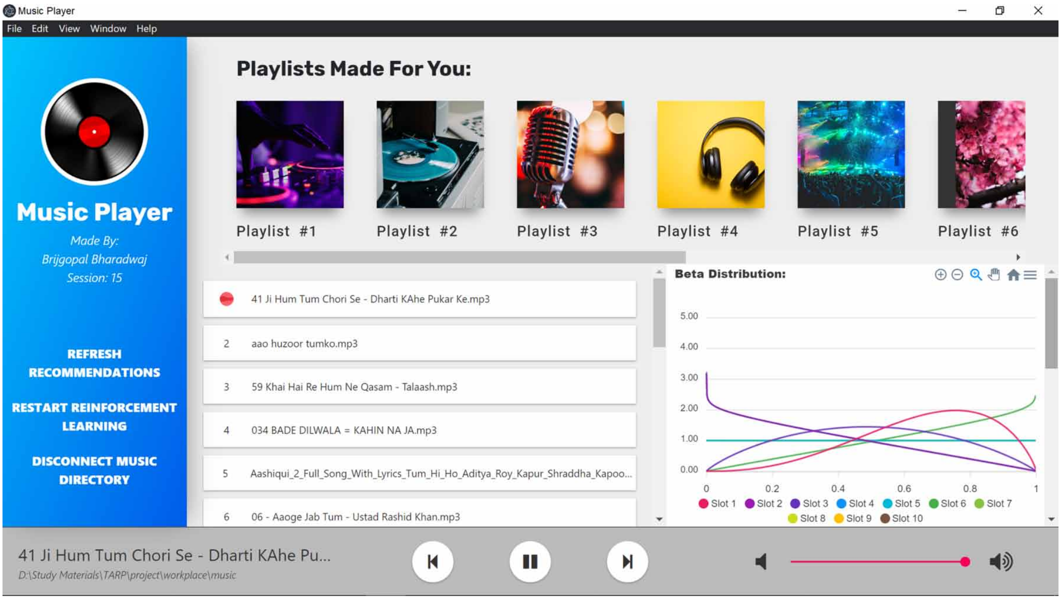


Table 1. Samples of Obtained Playlists

Playlist 1	Playlist 2
Broken – Jake Bugg	Old town road – Lil NasXft. Billy R...
A song about love – Jake Bugg	My Pony – Magic Mike XXL
Two fingers – Jake Bugg	Without Me – Eminem
Better – Khalid	Heart Attack – Enrique
Stitches – Shawn Mendes	Laila Mein Laila – Raees
Touch – Pia Mia	Worth It – Fifth Harmony
Lost in Japan (Remix) – Zedd& S.M.	Rockabye – Clean Bandit ft. Sean Paul
Delicate – Taylor Swift	Bailando – Enrique
Bloom – Troye Sivan	Honey, I'm good – Andy Grammer
Red – Taylor Swift	Swalla – JasonDerulo ft. NickyMinaj
Fools – Troye Sivan	Just my type – The Vamps
You belong with me – Taylor Swift	Toxic – Britney Spears

To gauge the performance of the presented algorithm, the system interaction quality index was also defined. As a part of its definition, different weight attributes were assigned to the various actions available to the user. These weights, over the period of time, provided a cumulative performance overview of the system, as experienced by each user individually. This user-interaction with the system was used as the metric to quantify the performance of the three implementations, and the values recorded over-time are presented in Fig. 12.

The analysis of the accumulated data presents the following patterns:

1. As the system keeps having interactions with the user, it modifies its constituent parameters, and hence the beta-distributions of the available choices, in such a way that the user-preferences are convincingly modelled by the selection algorithm.
2. The discounting effect ensures that the system keeps evolving and does not get locked in its recommendations, even after considerable time has elapsed. Fig. 13 presents a comparison of the beta distributions of the same choice, at different sessions.
3. The user-interaction with the presented algorithm presents a better user-experience then the two benchmarks, as it is in a constant pursuit of the balance between the explorative and exploitative approaches for recommendation.

CONCLUSION

In this paper we have presented the design and implementation of a music recommendation system, which uses the concepts of Bayesian reinforcement learning in a multiple-play environment. It aims at providing better recommendations to the user, by incorporating certain algorithms that account for the subjective nature of user preferences and their evolution over the course of time. The implementation is also made to work with a library of offline songs, and its performance during user-interaction is compared to explorative and exploitative benchmarks. In future, we aim to further test the performance of the proposed architecture by comparing it to the state of the art algorithms to obtain insights about their relative effectiveness.

CONFLICT OF INTEREST

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Figure 12. Temporal Progression of the System Interaction Quality Indexes

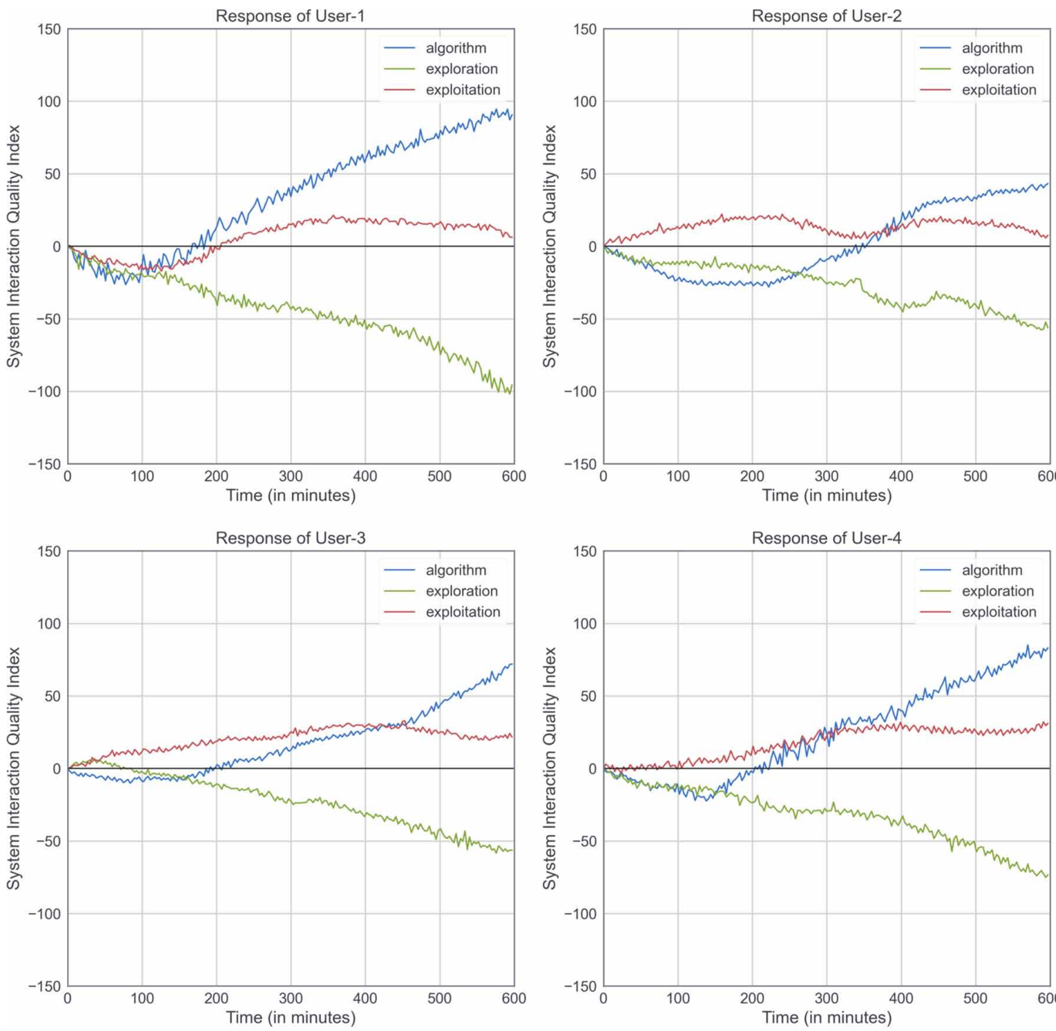
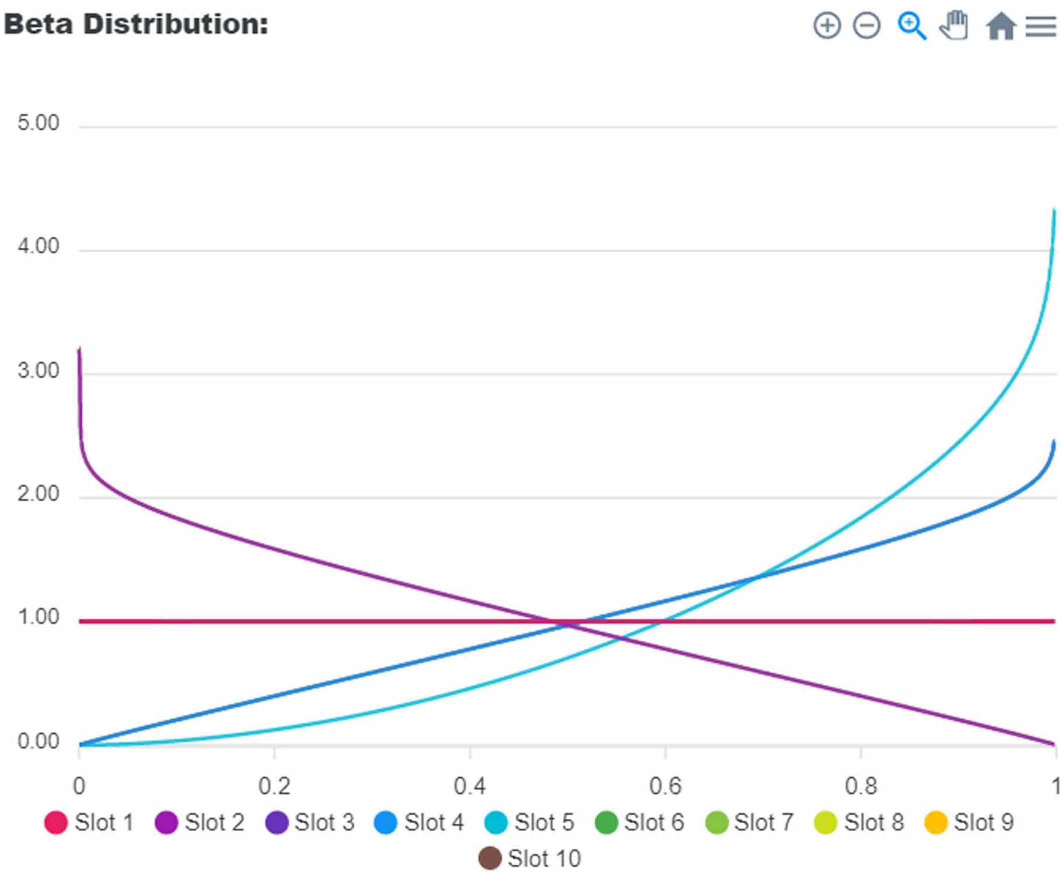


Figure 13. Selection Beta Distributions for a given choice, (a) during session 15 (b) during session 55



REFERENCES

Ellis, D. (2007). *Chroma feature analysis and synthesis.* Resources of Laboratory for the Recognition and Organization of Speech and Audio-LabROSA.

Harte, C., Sandler, M., & Gasser, M. (2006). Detecting harmonic change in musical audio. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia* (pp. 21-26). doi:10.1145/1178723.1178727

Kohli, P., Salek, M., & Stoddard, G. (2013). A fast bandit algorithm for recommendation to users with heterogenous tastes. *Proceedings of the AAAI Conference on Artificial Intelligence*, *27*(1), 1135–1141.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30–37. doi:10.1109/MC.2009.263

Liang, D., Zhan, M., & Ellis, D. P. (2015). Content-Aware Collaborative Music Recommendation Using Pre-trained Neural Networks. In ISMIR (pp. 295-301). Academic Press.

Louëdec, J., Chevalier, M., Mothe, J., Garivier, A., & Gerchinovitz, S. (2015). A multiple-play bandit algorithm applied to recommender systems. *The Twenty-Eighth International Flairs Conference*, 67-72.

Murthy, Y. S., & Koolagudi, S. G. (2018). Content-based music information retrieval (cb-mir) and its applications toward the music industry: A review. *ACM Computing Surveys*, *51*(3), 1–46. doi:10.1145/3177849

Radlinski, F., Kleinberg, R., & Joachims, T. (2008). Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on Machine learning* (pp. 784-791). Academic Press.

Sánchez-Moreno, D., González, A. B. G., Vicente, M. D. M., Batista, V. F. L., & García, M. N. M. (2016). A collaborative filtering method for music recommendation using playing coefficients for artists and users. *Expert Systems with Applications*, *66*, 234–244. doi:10.1016/j.eswa.2016.09.019

Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, *25*(3/4), 285–294. doi:10.2307/2332286

Uchiya, T., Nakamura, A., & Kudo, M. (2010, October). Algorithms for adversarial bandit problems with multiple plays. In *International Conference on Algorithmic Learning Theory* (pp. 375-389). Springer. doi:10.1007/978-3-642-16108-7_30

Wang, X., & Wang, Y. (2014, November). Improving content-based and hybrid music recommendation using deep learning. In *Proceedings of the 22nd ACM international conference on Multimedia* (pp. 627-636). doi:10.1145/2647868.2654940

Wang, X., Wang, Y., Hsu, D., & Wang, Y. (2014). Exploration in interactive personalized music recommendation: A reinforcement learning approach. *ACM Transactions on Multimedia Computing Communications and Applications*, *11*(1), 1–22. doi:10.1145/2623372

Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys*, *52*(1), 1–38. doi:10.1145/3285029

Brijgopal Bharadwaj received his B. Tech. in computer science engineering from the Vellore Institute of Technology, Vellore, India in 2021. His research interests include the applications of reinforcement learning to music-recommendation and audio analysis systems, and the application of parallel and distributed computing paradigms in the security domain, along with genetic optimizations for hyper-parameter tuning and enhanced performance. He has published a few papers in International Journals and conferences on parallel computing.

Ramani Selvanambi is an Assistant Professor (Senior) in the School of Computing Science and Engineering at Vellore Institute of Technology (VIT), Vellore, India. He received his B.E in Computer Science and Engineering from MNM Jain Engineering College, Madras University and M. Tech Computer Science and Engineering from Bharathidasan University and Ph.D. in Computer Science and Engineering from Vellore Institute of Technology (VIT), Vellore. He has 12+ years of experience in teaching and 2 years of experience in the Consultancy and Software Industry. He has published about 30 papers in International Journals on Machine Learning, Nature-Inspired Algorithms, Cryptography and Health Care. His research interest includes Data Mining, Machine learning, Database Systems, Optimization Techniques. He is a member of the Computer Society of India (CSI), IEEE and other technical societies.

Marimuthu Karuppiah received his B.E. degree in Computer Science and Engineering from Madurai Kamaraj University, Madurai, India in 2003, M.E. degree in Computer Science and Engineering from Anna University, Chennai, India in 2005, Ph.D. degree in Computer Science and Engineering from VIT University, Vellore, India in 2015. Presently, he is a Professor in Department of Computing Science and Engineering, SRM University, Delhi-NCR, India. Before joining SRMIST, he was with the VIT University, Vellore, India as an Associate Professor. He has published more than 40 research papers in SCI indexed journals. Also, he has published more than 40 research papers in SCOPUS indexed journals and international conferences. He is a life member of Cryptology Research Society of India (CRSI) and Computer Society of India (CSI), senior member of IEEE and member of ACM. His main research interests include cryptography and wireless network security, in particular, authentication and encryption schemes.

Ramesh Chandra Poonia is an Associate Professor at the Department of Computer Science, CHRIST (Deemed to be University), Bangalore, India. Recently completed his Postdoctoral Fellowship from CPS Lab, Department of ICT and Natural Sciences, Norwegian University of Science and Technology, Ålesund, Norway. He received his Ph.D. degree in Computer Science from Banasthali University, Banasthali, India in July 2013. His research interests are Sustainable Technologies, Cyber-Physical Systems, Path Planning & Collision Avoidance in Artificial Intelligence, and Intelligent Algorithms for Autonomous Systems. He is Chief Editor of TARU Journal of Sustainable Technologies and Computing (TJSTC) and Associate Editor of the Journal of Sustainable Computing: Informatics and Systems, Elsevier. He also serves in the editorial boards of a few international journals. He is the main author and co-author of 06 books and an editor of more than 25 special issue of journals and books including Springer, CRC Press – Taylor and Francis, IGI Global and Elsevier, edited books, and Springer conference proceedings. He has authored/co-authored over 65 research publications in peer-reviewed reputed journals, book chapters, and conference proceedings.