Mood-Adapted Content-Based Filtering Model for a Music Recommendation System

Abish Issagulov
KAIST, School of Computing
291 Daehak-ro, Yuseong-gu, Daejeon
issagulov@kaist.ac.kr

Jun Seong Kim
KAIST, School of Computing
291 Daehak-ro, Yuseong-gu, Daejeon
09jkim@kaist.ac.kr

SeungAn Jung
KAIST, School of Computing
291 Daehak-ro, Yuseong-gu, Daejeon
specialjsa@kaist.ac.kr

SeungWan Kang
KAIST, School of Computing
291 Daehak-ro, Yuseong-gu, Daejeon
seungwanwin@kaist.ac.kr

Washik Uddin Ahmed Mollah
KAIST, School of Computing
291 Daehak-ro, Yuseong-gu, Daejeon
washikuddin@kaist.ac.kr

Abstract

Music is an important component of the development of human culture and often holds itself as an identifying or character-building factor for many individuals. With the advent of technology, recommendation systems for music have become commonplace for the average listener. However, unfortunately, many of these systems or platforms often cannot fulfill the emotional requirements of individuals; instead often opting more for filtering or history-based algorithms in comparison to user mood-based recommendation systems [10]. In this paper, we propose a modified mood-adapted content-based filtering model to provide a recommendation system using the Spotify API based on a decision-tree mood-labeled song dataset along with a dataset of users and details about the songs that they have listened to. We were able to develop a mood-adapted content-based filtering model that was successfully able to match the accuracy of more traditional models, thus able to show that we were able to incorporate adaptive factors into content-based filtering models without a loss in performance.

1. Introduction

Music, a timeless and integral element of human culture, has played an important role in shaping both societal norms and the identities of individuals. Its impact extends beyond just entertainment, often being a powerful means of expression of emotional state and connection. In the modern digital world, new technology has changed how we listen to music, with apps recommending songs we might like.

Music is a subjective and emotional experience, and while most music recommender systems excel in leveraging user histories and preferences, a notable gap exists concerning their ability to address the emotional nuances inherent in music appreciation. As noted by Rentfrow [11] and Gosling [7] (2003), music preferences are intrinsically tied to an individual's personality and can be used to make judgments about their character and temperament. Therefore, a simple comparison of songs using metadata and user behavior history is not sufficient for comprehensive music recommendations.

To address this emotional aspect, new human-centered approaches like an emotion-based model and context-based model have been proposed. By considering affective and social information, these models largely improve the quality of recommendations as stated by Yading Song et al., 2017 [14]. The emotion-based model helps in identifying emotional states that can be linked to specific song features, such as tempo, lyrics, or harmony. On the other hand, the context-based model understands the environment in which users are listening and can identify mood shifts based on music selections.

Despite the advantages of these models, some challenges remain in implementation or evaluation. For example, establishing an objective evaluation method for music recommendation systems is still a complicated process. Most of the evaluation techniques are based on subjective system testing, wherein users rank different systems based on the playlist generated by various approaches [12]. According to Yading Song et al., 2017 [14], this subjective nature is one of the major drawbacks of evaluating recommender systems in music. Additionally, the availability of data and computational resources are also the major challenges that could hinder the implementation of more advanced algorithms to address this gap in music recommender systems.

Music recommender systems need to address the emotional nuances inherent in music appreciation, which could recommend suitable songs to the users. Context-based models and emotion-based models, both of which take into account the affective and social aspects of music enjoyment, are promising approaches for capturing these emotional nuances. However, there exist challenges with implementing and evaluating these models, such as data availability and the subjective nature of evaluations, which could hinder their widespread adoption.

1.1. Motivation

With the vast array of music that is now available freely online, there is a growing need for a more personalized music experience for users who often may find it challenging to discover new songs that align with their tastes and moods. As such, there have been several attempts and developed models that have aimed to recommend personalized songs to users, starting from collaborative systems-based approaches in the 90s with the introduction of models like Ringo [13] and music content-based approaches [15]. Contemporary and more sophisticated models have either expanded on these initial techniques or created hybridized methods.

However, many models that are currently in use are often generalized to be able to appeal to a wider user base. Our motivation for this study was due to our importance in understanding the emotional factors that humans have which can intrinsically affect what sort of music that they would want to listen to.

While many existing emotion-based models match the moods of songs to the current mood of listeners, we wanted to expand on this with a modified hybrid recommendation model that provides songs to users based on the trends of songs that they listen to under a certain mood rather than simply linking song moods to user moods. Such models will be able to provide users with songs that match their current mood and also take into account their listening history to give more personalized recommendations.

We realized some of the shortcomings that current content-based filtering methods have. They rely on the extraction of content features in music, such as its musical content and its features, and find its relation to the number of times a user may have listened to a song with certain weights attached to each user to indicate their preference for certain features [3]. This means that mood is not incorporated into such a model, but rather it is only based on how many times a user listened to a particular song. To fulfill our motivation to incorporate user mood into our recommendation system, we decided to modify this content-based model to include varied mood weights for each user as well.

2. Methodology

2.1. Mood Based Dataset [5]

In this section, we address the issues regarding data selection and processing. We begin with a thorough description of the dataset and then go on to explain our labelling strategies. The Spotify dataset has 19 features; the features are: valence, year, acousticness, artists, danceability, duration_ms, energy, explicit, id, instrumentalness, key, liveness, loudness, mode, name, popularity, release_date, speechiness, tempo and mood_prediction. There are 170,653 songs overall from 34088 artists. It contains newly released songs and old songs; that were released from the year 1921 to 2020. We used one hot encoding for representing the non-numeric features.

Not all of these features are equally important for predicting the mood of the song. Large models suffer from a problem called the curse of dimensionality, which means the performance of the model is adversely affected by the large dimension of the input data. Therefore it is essential for us to carefully select them, so as not to underfit the model or to suffer from high dimensionality.

2.1.1 Feature selection

We want to objectively decide how many features to select. For that, we used Recursive Feature Elimination [17] (RFE). RFE helps us choose the number of features that are most relevant to predicting a feature and also the model used to predict the feature. A brief description of RFE follows.

RFE operates as a wrapper-type feature selection algorithm, distinguishing itself by incorporating a distinct machine learning algorithm at its core. Unlike filter-based feature selection methods that evaluate each feature independently based on a score, RFE actively involves the selected machine learning algorithm in the process of feature selection.

Essentially, RFE functions as a wrapper-style feature selection algorithm with an inherent reliance on filter-based feature selection mechanisms. The algorithm initiates its search for a subset of features by commencing with all features present in the training dataset, and systematically eliminating features until the desired number is reached.

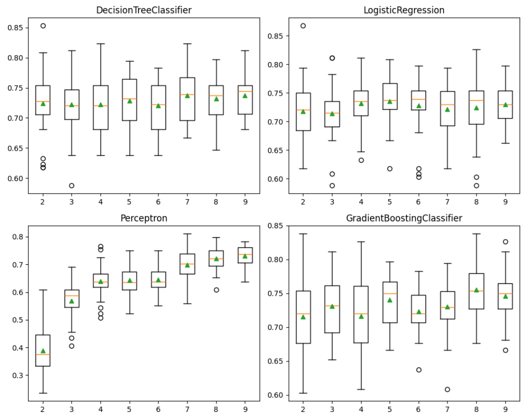

Figure 1. Model comparisons for number of features

The procedure involves fitting the designated machine learning algorithm, ranking features by their importance, discarding the least important ones, and subsequently refitting the model. This iterative process continues until the specified number of features is retained.

Figure 1 compares the number of optimal features for mood prediction according to different models. It is difficult to reason about which model to use for feature selection. Therefore, we have tried it with a decision tree classifier, logistic regression, perceptron and gradient boosting classifier. None of the models show any clear trend about the number of features, except perceptron which gives higher accuracy with more number of features. This result is problematic, since it does not give us an optimal number of features, and therefore opens the door for further exploration and manual feature selection.

Firstly we select out some features based on intuition. Such as album, song write, etc will lead to overfitting and therefore deemed unnecessary. Then we find the correlation coefficient of each of the remaining features with the moods, to see how strongly the features affect the mood. From Figure 2 we can see that some features have a very low correlation with the mood. Namely, key, popularity, and time_signature seem to have very low correlation with all the moods, therefore we do not include those in our training.

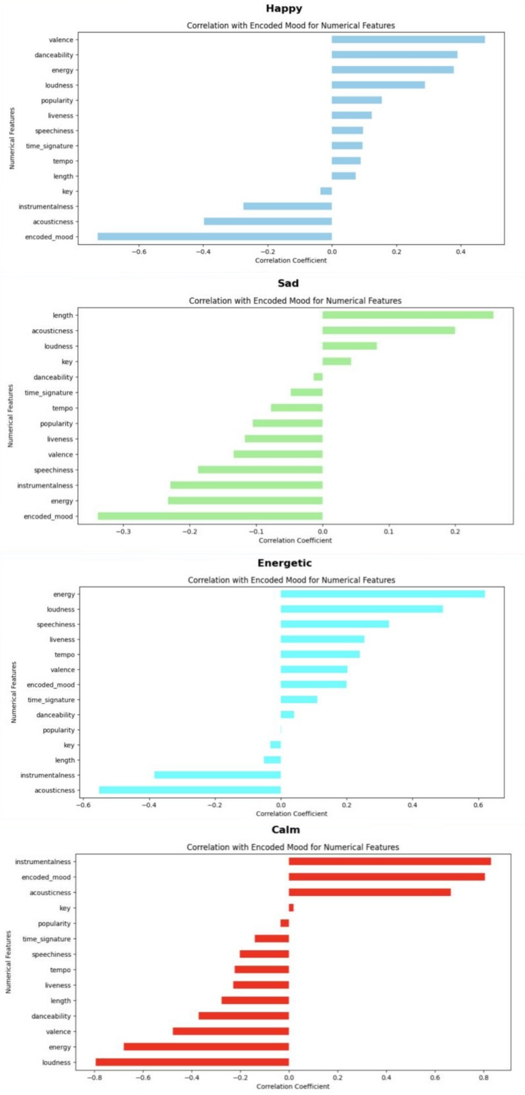

Figure 2. Correlation coefficient between mood and song features

Mood	Count
Sad	81728
Happy	47404
Energetic	21457
Calm	20064

Table 1. Count of Each Mood

2.1.2 Labelling

We decided to use the decision tree classifier for predicting. We select nine as the number of features. The RF algorithm picks the nine most significant features and trains the decision tree [2]. Decision trees are powerful models that leverage a tree-like structure to make decisions or predictions by recursively splitting the dataset based on feature conditions, allowing for a comprehensive analysis of complex relationships within the data. Table 1 shows that the count of moods in the dataset is skewed. It is worth pointing out that the predictions differ slightly based on the model we use; this is one unresolved issue in our paper.

2.2. Recommendation System

2.2.1 Dataset

Both Collaborative Filtering and Content-Based Filtering require a dataset with listening counts that can represent a user's preference for music. We obtained a dataset of 10000 unique songs with 76353 unique users, which also contains information about how many times a user listened to a specific song within the dataset. (https://www.kaggle.com/datasets/anuragbanerjee/million-song-data-set-subset) Accordingly, we have built a user-item matrix to create filtering data for the recommendation system.

2.2.2 Mood-Adapted Content-Based Filtering

Among the typical methods of recommendation systems, the most used method is collaborative filtering, which is a method that can predict how much a specific user may have a preference for a new item through a preference correlation between multiple users and multiple items.

Among them, model-based collaborative filtering uses machine learning algorithms to predict the preferences of items that users have not yet evaluated. It is based on the idea that a user's preference can be determined by latent factors for users and items. In this way, the user-item estimated matrix can be extracted by using a matrix decomposition such as SVD decomposition [9] that predicts the preference for a new item.

In the case of a music recommendation system, there is a problem referred to as ”cold start”: it fails when no usage data is available, so it is not effective for recommending new and unpopular songs. [16]. As such, a content-based recommendation can be more efficient due to the clarity of the features of each item(song). Therefore, we adopted content-based filtering as our chosen methodology for music recommendation.

The next process involved making recommendations based on the mood of each song. Since the existing recommendation system does not make recommendations based on the mood of the music the user wants to listen to, we made this possible by adding a latent-set matrix that varies depending on different moods. A particular mood can be selected with a one-hot encoded mood vector and added to the user-latent vector. Thus, we refer to our model as a mood-adapted content-based filtering.

Figure 3 illustrates the overall structure of mood-adapted content-based filtering. Unlike content-based filtering originally considered only the user's latent vector p and the fixed song's feature q, we now keep the latent vector in the base and add one more vector set matrix that varies according to the mood. In this way, p becomes a term that considers the user's preference for all songs, and each M_i vector becomes a term that is reflected only in the user's specific mood condition.

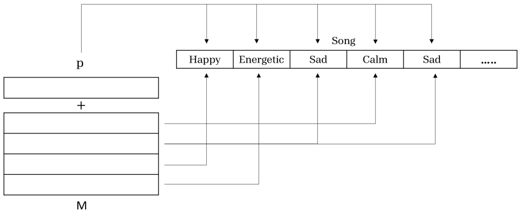

Figure 3. Mood-Adapted Content-Based filtering

2.2.2.1 Base Approach

Spotify provides 14 features for songs through its API. [6] As described in Section 2.1.1, we extracted a 9-dimensional song feature vector from this data to extract mood. Let p be the vector to represent the user's latent vector and q be the feature vector for the song. Let M be the 9*4 matrix that is selected differently row-by-row depending on mood and vector v is a one-hot encoding vector to represent the mood. Let C be the number of times the user listened to the song with feature q. Then the loss function can be defined as follows:

$$L = ||C - q^T(p + Mv)||_2^2 \tag{1}$$

2.2.2.2 Adding Neural Network

If we simply set the dimension of the latent vector p to the number of song features, the number of dimensions is small, so even if we filter by inner product, the estimation is inaccurate. To solve this problem, we added a fully connected layer to the mood-adaptive filtering to increase the number of variables to be learned, and when we evaluated it after applying it, we could see that the error rate of the test set was noticeably reduced. Our final loss function to calculate the process of content-based filtering with a neural network can be represented as follows.

$$L = ||C - NN(q, p + Mv)||_2^2 \tag{2}$$

We calculate the loss by weighting the variation over the set of M matrices by the latent vector p to measure the preference of a particular user for a new song. Filtering is performed by minimizing the loss at each epoch, and the overall mood-adaptive content-based filtering can be expressed as follows.

$$(p^*, M^*) = \underset{p,M}{\operatorname{argmin}} \sum_i L(p, M, v, q_i) \tag{3}$$

2.2.3 Evaluation

Since a small listening count for a particular song could indicate that a user listened to and decided that a song was not of their preference, we created the assumption to interpret this as a low preference for the song by a specific user. We then sorted the user's song listening counts in such a way that half of the songs were the user's preferred songs, while the other half was not the user's preferred songs, from our previously stated assumption.

We predicted the listening counts of songs in the test set according to the learned p* and M*, sorted them, and measured the hit rate to see how many songs with a median of the predicted listening count or higher were included in the user's preferred song group.

Other metrics used by recommendation systems other than hit rate include root-mean-square error (RMSE) and Precision@K [4]. However, we decided that hit rate is a better metric than these methods due to the characteristics of the listening count being continuous rather than discrete, leading to an easier expression of how satisfied the user is with the recommendation system with the ratio of liked to disliked songs within the outputted recommended song list.

We can estimate the user's preference for the new song from p^*, M^* shown above. If the current user wants to listen to a song represented by the mood of v, the preference of the user for a song with the characteristics of q can be expressed as follows.

$$E_i = NN(q_i, p^*, M^*v) \tag{4}$$

The listening count is not discrete, as it is not a metric of whether the user likes a song or not, but rather continuous data of how many times a user listened to a song. Thus, rather than a metric like Precision@K, we normalized the listening count to the maximum number of times a user listened to a song and measured the accuracy as the error rate of the difference between the actual value and the predicted value.

$$Accuracy(\%) = (1 - |\overline{\frac{E_i - C_i}{C_i}}|) \times 100 \tag{5}$$

In addition, the dataset contains counts of how many times a particular user listened to each song, from this we sorted by count, and considered songs with counts above the medium listen count as perferred songs while those below as non-preferred songs. From the total set of user data, we selected 10% to be used in our testing phase. We defined hit rate to be the proportion of actually preferred songs within the predicted preferred songs of the test set. Letting C_i be the actual number of times the songs in each test set were listened to, we measured the Hit Rate of our recommendation system as follows.

$$HitRate = \frac{n(\{C_i > Med(C)\} \cap \{E_i > Med(E)\})}{n(\{E_i > Med(E)\})} \tag{6}$$

3. Evaluation

3.1. Code Explanation

To give weights to every feature of each song in a proportional manner, we first normalized all features. Within our dataset, the largest number of unique songs listened to by a single user was 600 songs. As we have limited data, we implemented 5-fold cross-validation. We also set the epoch count to 500 using mini-batch gradient descent with Adam optimization [8]. For the neural network, we structured a multi-layer perceptron with fully connected layers. We set the size of the 2 fully connected layers as (number of features)*(hidden size) and (hidden size)*1 respectively and the non-linear function as ReLu function [1]. For the loss function, we used mean squared error and for accuracy, we used distinction rate. Finally using testing data, we could obtain hit-rate.

We did this procedure for content-based filtering (Figure 4), content-based filtering with the additional neural network (Figure 5), and finally mood-adaptive content-based filtering with the additional neural network (Figure 6).

3.2. Results

By training, we were able to set the hyperparameters. For learning rate, 1e-3 was found to be suitable. By setting the hidden size as 20, we achieved the highest accuracy while avoiding overfitting.

As shown in the error rate, we can see that its value decreases with training across all three models. In the case of the first model, content-based filtering, we can see that the validation error does not decrease after a certain point, which can be interpreted as the limit of the model. Since it was trained with 10 trainable variables corresponding to each feature number, even if the optimal solution was found, the error was large due to the limitations of the model. For the second content-based filtering with the additional neural network, we were able to increase the size of the model through the neural network. As a result, we can see that the training error and validation error converge as shown in the graph. The final validation error of the content-based filtering with the neural network was 46.2, so we viewed this as a meaningful result. Finally, for the mood-adaptive content-based filtering with the neural network model, we can see that the validation loss converges just like the content-based filtering with the additional neural network. The final validation error was 51.3, which is similar to the second model. Thus, we can see that the model with mood adaptive can perform similarly to the existing model.

3.3. Hit Rate

We also measured the accuracy of the recommendation system during the training process by calculating the hit rate using the test data for every 50 epochs. As a result, we obtained the following hit rates for each model. (Figures 7,8,9)

In the case of the content-based model, the results were not high, as mentioned in the loss function. The hit rate obtained through the test data was 0.55, which is practically meaningless. In the case of content-based filtering with the neural network, the hit rate tended to increase. From this, we can infer that the process of learning with MSE has a significant relationship with the recommendation process. In the end, we achieved a result of 0.62. Finally, for the mood-adaptive version, we found that the hit rate tended to increase for each epoch, just like the content-based filtering with the neural network. The final result is also 0.592, which shows that the mood-adaptive model can be as accurate as the traditional model.

3.4. Result Analysis

In the case of the most basic method, the content-based model, we could see the limitations of the model due to limited variables, and in the case of the neural network model, which we devised to solve this problem, we were able to design a model that can reduce the loss while adjusting the size of the model and prevent overfitting. We were able to confirm that the neural network had a significant impact on the performance improvement in the hit rate, and this showed that the recommendation model through the neural network was suitable. Under these conditions, we compared the performance of the model with mood-adaptive and the existing neural network. In the case of performance comparison, the mood-adaptive model considers the mood of the song as the current mood when calculating the preference of the song. In conclusion, we found that the two models performed similarly, indicating that the mood-adaptive model is also suitable for recommendation models.

In the case of the mood-adaptive model, more variables need to be trained as the variable related to mood is added to the existing model. In addition, it can be assumed that randomness plays a larger role than in the conventional model because the variables to be learned are selected according to the data in the batch. This characteristic can also be seen in the unstable behavior of the graphs. It was possible to solve this problem to some extent by increasing the data and batch size, we can extrapolate this behavior to conclude that this problem can further be solved with more data.

In the case of music recommendation by selection of the desired mood, we could see that different moods produced different results even for the same person. We could also see that songs other than the selected mood could be displayed when calculated according to individual preferences.

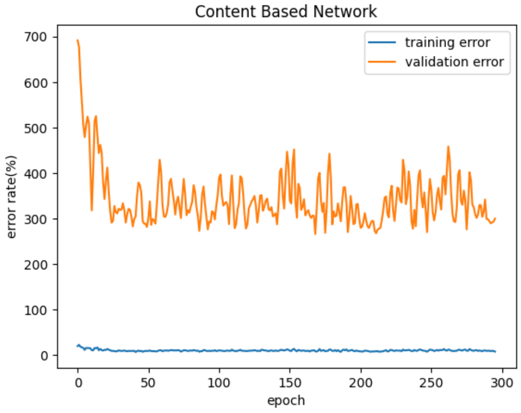

Figure 4. Accuracy - Content-Based filtering

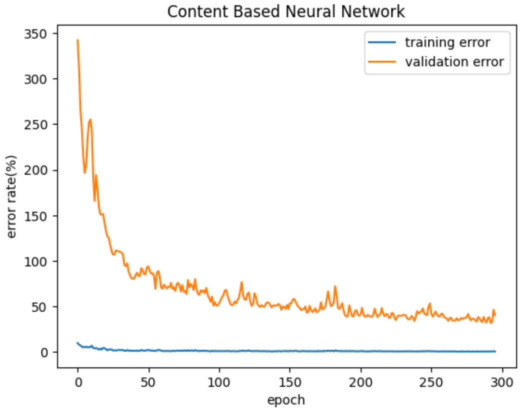

Figure 5. Accuracy - Content-Based filtering with additional Neural Network

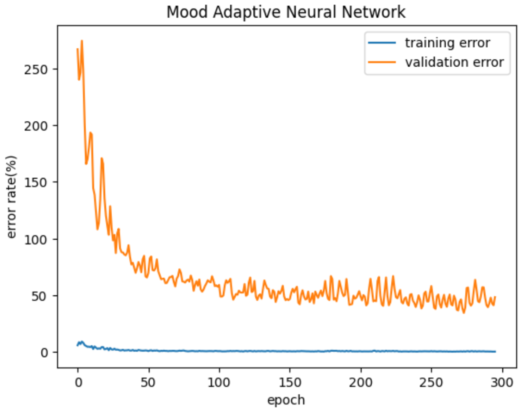

Figure 6. Accuracy - Mood-Adapted Content-Based filtering with additional Neural Network

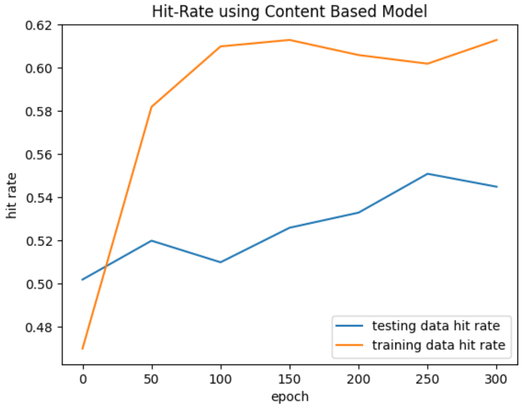

Figure 7. Hit Rate - Content-Based filtering

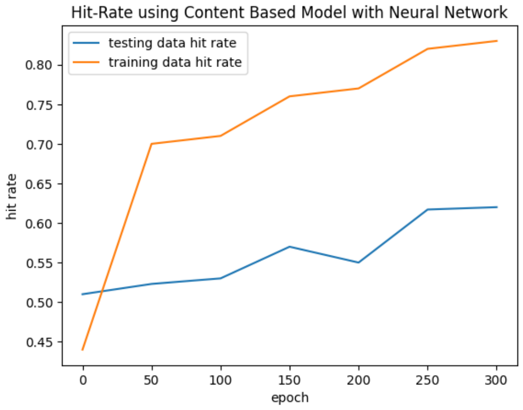

Figure 8. Hit Rate - Content-Based filtering with additional Neural Network

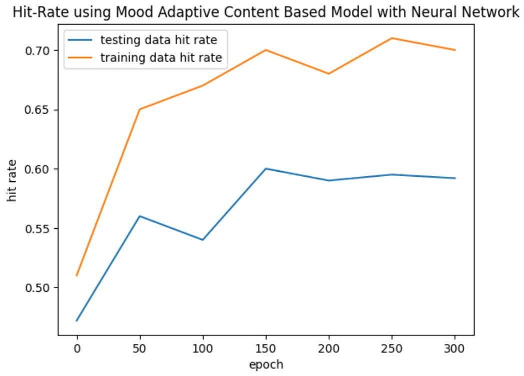

Figure 9. Hit Rate - Mood-Adapted Content-Based filtering with additional Neural Network

4. Conclusion

4.1. Summary of the research process

It was expected that content-based filtering would produce results that better reflect the user's preference as the features of the song are considered more than collaborative filtering.

The challenge of how well the recommendation system can reflect the mood of the song that the user wants was addressed by devising a new content-based filtering method that can be adjusted to learn according to each mood condition by reflecting the variance by mood in the user-latent vector.

Since the dataset is not discrete data regarding whether the user likes the song or not, but continuous data of how many times the user listened to the song, we measured accuracy as the error rate of the difference between the actual value and the predicted value after normalizing the listening count to the maximum number of listens, rather than a metric such as Precision@K.

Apart from accuracy, we also measured hit rate by considering songs with a song listening count greater than the median value as preferred songs and songs with a song listening count less than the median value as non-preferred songs.

4.2. Summary of findings

In the case of Accuracy, we can see that the error rate is over 300% in the early epochs of learning which then converges to 50

In the case of hit rate, we can predict that it easily exceeds 50% even with random picking as we can see that the songs in the dataset have a ratio of 50:50 favorite to non-favorite respectively. In our model, the hit rate was around 70, which we can conclude is significant.

In the end, it was verified that our new filtering method, which considers the challenging part described above, can be used in a recommendation system. We expect that the results of this study can be used as a base method for complex recommendation systems with continuous data, where the results should be varied according to the changing requirements of users.

References

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu), 2019. 5

[2] Mohammed Awad and Salam Fraihat. Recursive feature elimination with cross-validation with decision tree: Feature selection method for machine learning-based intrusion detection systems. *Journal of Sensor and Actuator Networks*, 12(5), 2023. 3

[3] Marine Chemeque Rabel. *Content-based music recommendation system : A comparison of supervised Machine Learning models and music features*. PhD thesis, 2020. 2

[4] Yeounoh Chung, Noo-ri Kim, Chang-yong Park, and Jee-Hyong Lee. Improved neighborhood search for collaborative filtering. *INTERNATIONAL JOURNAL of FUZZY LOGIC and INTELLIGENT SYSTEMS*, 18:29–40, 03 2018. 5

[5] Marvin Ray Dalida, Lyah Bianca Aquino, William Cris Hod, Rachelle Ann Agapor, Shekinah Huyo-a, and Gabriel Sampedro. Music mood prediction based on spotify's audio features using logistic regression. pages 1–5, 12 2022. 2

[6] Deniz Duman, Pedro Neto, Anastasios Mavrolampados, Petri Toiviainen, and Geoff Luck. Music we move to: Spotify audio features and reasons for listening. *PLOS ONE*, 17(9):1–18, 09 2022. 4

[7] Samuel D Gosling, Peter J Rentfrow, and William B Swann. A very brief measure of the big-five personality domains. *Journal of Research in Personality*, 37(6):504–528, 2003. 1

[8] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 5

[9] V. Klema and A. Laub. The singular value decomposition: Its computation and some applications. *IEEE Transactions on Automatic Control*, 25(2):164–176, 1980. 4

[10] Dip Paul and Subhradeep Kundu. *A Survey of Music Recommendation Systems with a Proposed Music Recommendation System*, pages 279–285. 01 2020. 1

[11] P. J. Rentfrow and S. D. Gosling. Message in a ballad: The role of music preferences in interpersonal perception. *Psychological Science*, 17(3):236–242, 2006. 1

[12] Paul Resnick, Hal R. Varian, and Guest Editors. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997. 2

[13] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating word of mouth. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co. 2

[14] Yading Song, Simon Dixon, and Marcus Pearce. A survey of music recommendation systems and future perspectives. 06 2012. 1, 2

[15] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002. 2

[16] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. 4

[17] Xiangyan Zeng, Yen-Wei Chen, and Caixia Tao. Feature selection using recursive feature elimination for handwritten digit recognition. In *2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 1205–1208, 2009. 2