JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

Music Recommender System Based on Play Count Using Singular Value Decomposition++

Muhamad Elang Ramadhan, Agung Toto Wibowo*

School of Computing, Informatics, Telkom University, Bandung, Indonesia
Email: [1]melangramadhan@student.telkomuniversity.ac.id, [2,*]agungtoto@telkomuniversity.ac.id
Correspondence Author Email: agungtoto@telkomuniversity.ac.id

Abstract−The availability of digital music content on various music streaming services, which is constantly growing, has increased the need for recommender systems (RS) to assist users in finding music that suits their taste without the need of searching manually. One of the commonly used paradigms is Collaborative Filtering (CF). In CF, the input used to predict ratings can take the form of explicit or implicit input from user feedback. In the music domain, implicit feedback such as the number of music plays can be utilized to predict a user's music preferences. Singular Value Decomposition++ is one of the Matrix Factorization (MF) algorithms that can leverage implicit feedback and address the sparsity issue. In this research, a music recommender system is built using the Million Song Dataset (MSD) Subset from The Echo Nest, utilizing SVD++ algorithm. Additionally, the performance of the built system is measured through k-fold cross-validation using the evaluation metrics RMSE and NDCG. The performance measurement results using RMSE and NDCG in 5-fold cross-validation yield an RMSE of 0.4423, NDCG@5 of 0.8232, and NDCG@10 of 0.8231 for the top 10 items.

Keywords: Recommender System; Collaborative Filtering; Singular Value Decomposition++; Music

1. INTRODUCTION

With the ease of accessing music streaming services such as JOOX, Spotify, and Apple Music, the quantity of digital music in this era will continue to grow. With this rapid growth, the task of discovering new and suitable music for users can lead to decision fatigue, and a tendency to defer making choices when confronted with an overwhelming number of choices [1]. Recommender systems (RS) can help users overcome these problems by providing them with a set of music recommendations that suit their preferences [2]. Collaborative Filtering (CF) is among the used and successful paradigms for building an RS [3].

The main concept of the CF paradigm is to predict the ratings that users will give to each item based on the similarity between users and/or other items [3]. CF has two approaches: memory-based and model-based. Although frequently used, memory-based CF requires a lot of resources because the matrix needs to be loaded into main memory each time a recommendation is made [2]. Furthermore, issues such as synonymy, sparsity, and scalability [4] are limitations of this approach. In the model-based approach, the prediction process is based on a pre-built model using machine learning algorithms and data mining techniques [2], [5]. Although it requires time to build the model and train the data, the storage required is smaller, and it can address the issues of memory-based approaches, resulting in more accurate predictions [6].

Ratings used in this paradigm are divided into two types: explicit feedback and implicit feedback [4]. Explicit feedback is direct user input on an item, such as providing a rating, while implicit feedback is indirect, such as the number of plays, clicks, or transactions. Explicit ratings can also have implicit ratings, for example, when a user rates a news article, we can infer (implicitly) that the user likes the topic or the writer of that article [7]. Leveraging implicit feedback in recommender systems can increase user interaction compared to using explicit feedback [8]. One method that can be used to predict ratings is Matrix Factorization.

Matrix Factorization (MF) is a commonly used supervised learning approach that achieves high accuracy in addressing sparsity issues in the data used in RS [9]. Singular Value Decomposition (SVD) is one of the MF algorithms that works by reducing the dimensionality of the matrix [10]. Singular Value Decomposition++ (SVD++) is an extension of the SVD algorithm [5] and can utilize implicit feedback as input, which better reflects user preferences [11].

Tian et al. in 2019 [12] used Logistic Regression (LR), XGBoost, and a combination of both (LX) to build a music recommender system using the MSD dataset [13]. Before making predictions, user profiles are constructed based on their music playback history. The prediction results are evaluated using error metrics and Area Under Curve (AUC). LR yielded an error value of 0.3062 and AUC of 0.7268, XGBoost had an error of 0.2723 and AUC of 0.7663, and LX achieved an error of 0.2376 and AUC of 0.8087. Based on these results, it can be concluded that LX outperformed LR and XGBoost models.

Dong et al. conducted research in 2020 [14], the Last.fm dataset from MSD [13] was used to build a music recommender system based on Fusion Deep Learning model. The input features for the model included audio files converted into spectrogram images and lyrics rewritten without losing their emotional expressions. The model used feature extractors based on Convolutional Neural Network (CNN) for spectrogram and emotion extraction from lyrics, along with an LR-based recommendation prediction model. The prediction results were evaluated using accuracy metrics, and the recommendation accuracy was reported as 90.2.

In 2018, Liu et al. [15] conducted research using a book lending dataset over a four-year period, consisting of user ID, book ID, loan days, and timestamp from the Aleph library system in a school. The loan days feature,

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

an implicit feature, was used as input for the recommender system built using the SVD++ algorithm. Testing was performed with different numbers of latent factors (20, 50, 100, 150, 200, and 500), and the evaluation using RMSE showed that the model with 100 latent factors had the best accuracy with an RMSE value of 0.977.

Biswal et al. conducted research in 2020 [2] to build a music recommender system using the MSD [13] Subset from The Echo Nest, utilizing the Restricted Boltzmann Machine (RBM) and the CF engine by Apache with implicit feedback in the form of user music play counts as input. The dataset was divided into 80% training and 20% testing sets before evaluation. Confidence scores, ranging from 0 to 1, were used as ratings in the prediction phase, calculated based on the number of music played. Confidence scores represent the certainty of recommending an item to a user. The evaluation results using the Normalized Root Mean Squared Error (NRMSE) metric showed that RBM had better accuracy than the CF engine by Apache with value of 0.0075 and 0.0200 repectively.

Pujahari et al. in 2020 [16] conducted research to compare the performance of model-based CF methods, utilizing MovieLens dataset. One of the methods used was matrix factorization, with two algorithms employed: SVD and SVD++. The performance of the built models was measured using Normalized Discounted Cumulative Gain (NDCG) with a parameter D = 60, resulting in values of 0.4154 for SVD and 0.4441 for SVD++. From these results, it can be concluded that SVD++ outperformed SVD.

Research by Chen et al. in 2020 [17] built a recommender system using the MSD Last.fm dataset [13] with a hybrid approach consisting of item popularity-based method and SVD, utilizing the music play count feature. Before being used for prediction, these play counts were transformed into LScore ranging from 1 to 5. The main concept of LScore was to consider the ranking of music. The prediction results were evaluated using precision metrics, and the hybrid model that was built was compared with the results from the SVD model and item popularity. The hybrid model showed better prediction results compared to the other two models, with a precision value of 0.509, compared to 0.438 for SVD and 0.407 for item popularity in the top 4 music recommendations.

In this research, we will be developing a recommender system and performing an in-depth analysis of the recommender system model's performance built with SVD++ and Million Song Dataset The Echo Nest Taste Profile Subset dataset. In the development process, we utilize implicit feedback in the form of the number of music played as the input for the SVD++ model. Additionally, to differentiate with previous research we incorporated k-fold cross-validation with combination of RMSE and NDCG evaluation metric to evaluate the model. Through this research, we aim to offer valuable insights and establish a foundational benchmark than can be utilized for future research comparisons.

2. RESEARCH METHODOLOGY

2.1 Research Stages

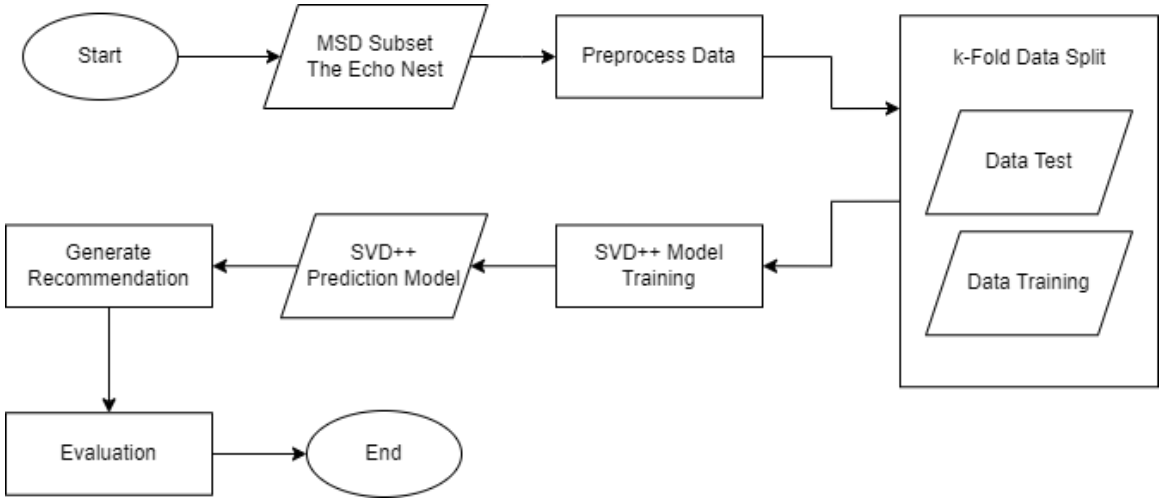

Figure 1. Recommender system development flow

As shown in Figure 1, this research begins with preprocessing the data we acquired, Million Song Dataset (MSD) The Echo Nest Taste Profile Subset by categorizing the play count feature. Then, the data is split into k-folds for k-fold cross-validation. Afterward, SVD++ model will be trained for later to be used in item recommendation generation for user. An overview of this process can be seen in Figure 1. Additionally, a detailed explanation for the process will be discussed in the next sections of this paper.

2.2 Singular Value Decomposition++

Singular Value Decomposition++ (SVD++) is an extension Singular Value Decomposition (SVD) algorithm, a model-based approach, and Matrix Factorization (MF) method. The main concept of this algorithm is to uncover

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

latent features from the low-rank approximation matrix generated after reducing the user-item matrix. In SVD++, implicit feedback is taken into consideration as supplementary information to tackle challenges related to sparsity and cold start problems [10].

SVD++ reduces high-dimensional user-item matrix by decomposing it into the product of two lower-dimensional matrices [18]. The prediction formula of the SVD++ algorithm can be seen in equation 1.

$$\tilde{r}_{ui} = b_{ui} + q_i^T \left(p_u + |N(u)|^{\frac{-1}{2}} \sum_{j \epsilon N(u)} y_j \right) \quad (1)$$

$$b_{ui} = \mu + b_u + b_i \quad (2)$$

where:
$\tilde{r}_{ui}$: predicted rating given by user u to item i
b_u : bias vector of user u
b_i : bias vector of item i
q_i : latent feature for item i
p_u : latent feature for user u
$N(u)$: set of the items for each user u with implicit preferences
y_j : characteristic vector of user u with implicit preferences towards item i

To mitigate overfitting during the training process, we minimize regularized squared error [18] with the following objective function:

$$\sum_{r_{ui} \epsilon R_{train}} {e_{ui}}^2 + \lambda(b_i^2 + b_u^2 + \|q_i\|^2 + \|p_u\|^2) \quad (3)$$

$$e_{ui} = \tilde{r}_{ui} - r_{ui} \quad (4)$$

To further optimize the objective function, we utilize Stochastic Gradient Descent (SGD) [18] in equation 3 with the following formula:

$$b_u \leftarrow b_u + \gamma(e_{ui} - \lambda b_u) \quad (5)$$

$$b_i \leftarrow b_i + \gamma(e_{ui} - \lambda b_i) \quad (6)$$

$$p_u \leftarrow p_u + \gamma(e_{ui} \cdot q_i - \lambda p_u) \quad (7)$$

$$q_i \leftarrow q_i + \gamma(e_{ui} \cdot p_u - \lambda q_i) \quad (8)$$

where:
γ : learning rate
λ : regularization term

Parameters b_u, b_i, p_u, dan q_i are updated using SGD with equations 5, 6, 7, and 8 until the gradient vector approaches near zero [18].

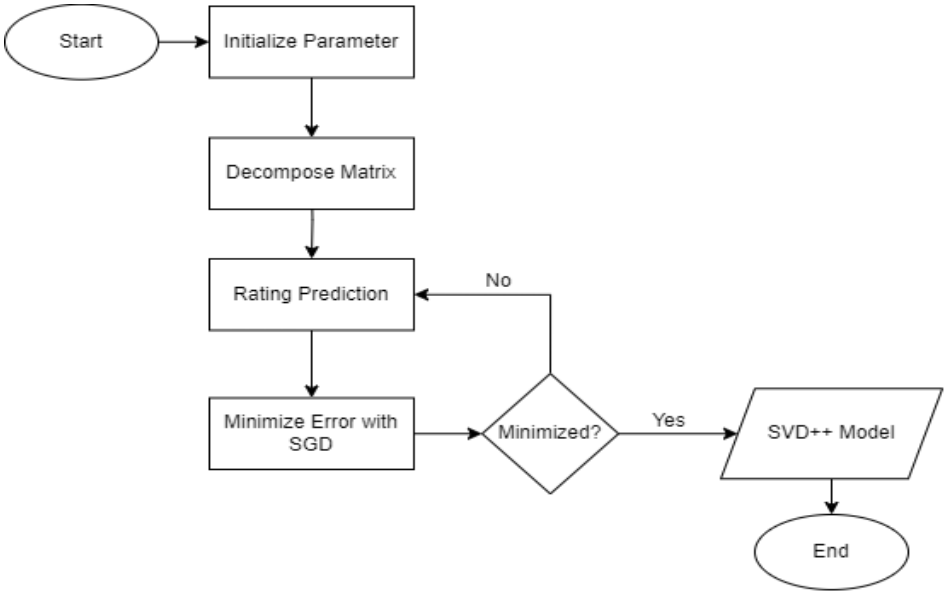

Figure 2. SVD++ model training flow

Based on Figure 2, the training of the SVD++ model begins by initializing parameters such as the dataset to be utilized, n_epochs, and n_factors. Subsequently, the process of matrix decomposition is executed to generate a low-rank approximation matrix, which is achieved by reducing the user-item matrix constructed from the dataset.

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

The latent features derived from this matrix are then employed to calculate the rating predictions using equation 1. To minimize the error in the prediction results, an objective function outlined in equation 3 is utilized in conjunction with the Stochastic Gradient Descent (SGD) technique, as depicted in equations 5, 6, 7, and 8. This iterative minimization process continues until the error reaches its minimal value, resulting in an SVD++ rating prediction model.

In this research, the modeling process is aided by the "surprise" library [19]. The utilization of this library helps in producing an optimal and efficient SVD++ model, resulting in more accurate item recommendations.

2.3 Recommendation Generation

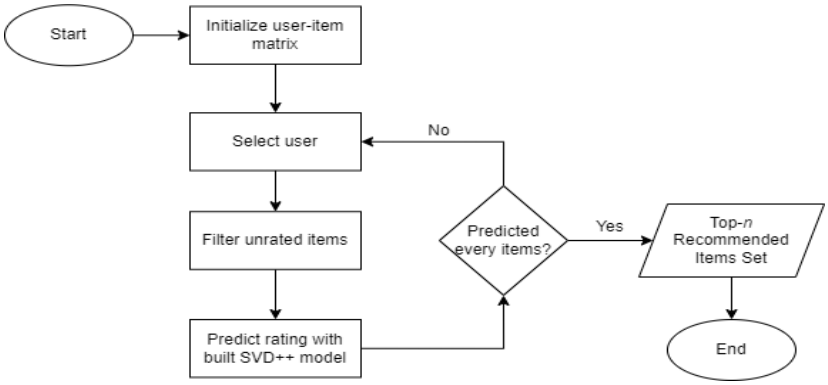

Figure 3. Recommendation generation flow

Based on Figure 3, an m × n user-item matrix is made to be utilized in generating recommendations. To generate these recommendations, a filtering process is applied to the matrix, selecting only those items that have not yet been rated by the user. Subsequently, the SVD++ model is used to predict ratings for these unrated items and store it inside the user-item matrix. The prediction process is performed for each unrated item, considering every user within the dataset. The model then outputs the top-n recommended items for the user.

Table 1. User-item matrix illustrations

User \ Item	FightSong – Eve	Payphone – Maroon 5	golden hour –JVKE
John	?	1	4
Jane	2	3	?
Michael	?	?	5
Sarah	5	5	1

Table 1 provides a visual representation of the user-item matrix, depicting the ratings assigned by users to all items within the dataset. In this matrix, a rating value denoted as "?" indicates that the user has not yet provided a rating or interacted with the corresponding item. These unrated values will be predicted using the prediction model.

2.4 K-fold Cross-Validation

The selection of the SVD++ model for generating recommendations will be determined through a process of k-fold cross-validation. In this research, we have opted for a k value of 5, which means that the dataset is divided into five folds, with each fold comprising 20% of the data for training and 80% for testing. This division allows us to evaluate the model's performance when using different datasets. For every iteration of the cross-validation process, testing is conducted using RMSE and NDCG.

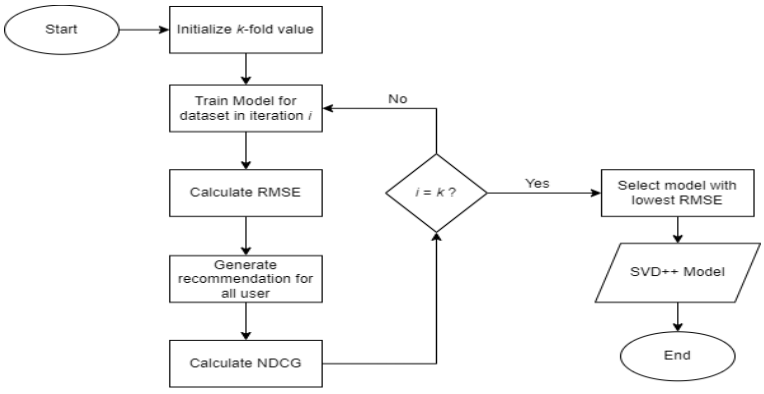

Figure 4. K-fold cross-validation flow

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

In this research, the flow of k-fold cross-validation is depicted in Figure 4. The process begins by setting the value of k-fold. Each fold iteration involves training the SVD++ model using a specific split of the dataset. The trained model is then evaluated by computing the RMSE score. Recommendations for every user will be generated to evaluate the recommendation ranking relevancy of the model by computing the NDCG for every user, and then calculating the average of the NDCG scores. After the process is done for k times, the best model will be selected with the criteria of the lowest RMSE.

Additionally, as part of our experimental approach, we explored the use of 10-fold cross-validation. This extended form of cross-validation provides even more robust validation by dividing the dataset into ten folds, with 10% of the data for training and 90% for testing in each fold. To facilitate the process of conducting cross-validation, we employed the "surprise" library, a powerful tool that streamlines the implementation of cross-validation and simplifies the evaluation of the SVD++ model's performance.

3. RESULT AND DISCUSSION

3.1 Dataset and Data Preprocessing

For the modeling of the recommender system, the dataset to be used is the MSD Subset The Echo Nest [13], consisting real data of user's music listening history. Each record in the dataset represents a unique instance of a user's interaction with a song.

Table 2. Dataset features

Name	Type	Description	Frequency
user_id	String	Unique identifier of user	76,353
music_id	String	Unique identifier of music	10
play_count	Int	User's play count of music	2,000,000

Based on Table 2 above, this dataset consists of 76,353 users, 10,000 songs, and 2,000,000 user play counts of songs, with features including user ID, music ID, and play count.

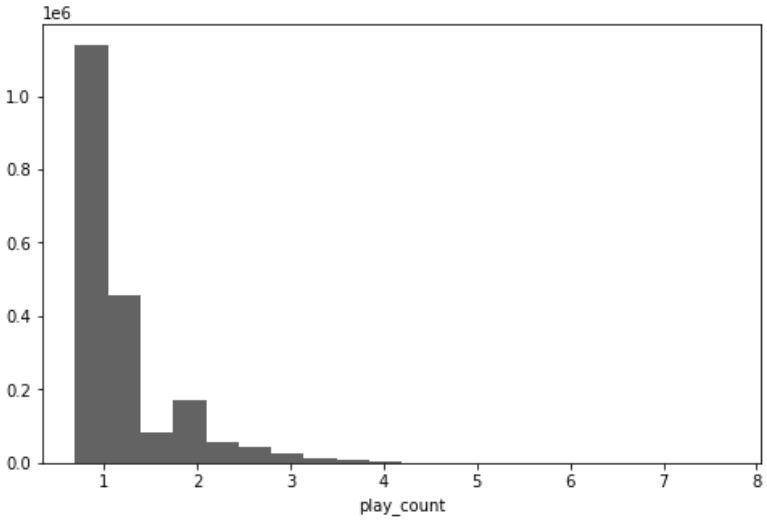

Figure 5. Log scaled play_count value distribution

Based on Figure 5, the distribution of play_count in this dataset feature values tend to be skewed to the left, with the smallest value being 1 and the largest being 2213. This can result in the built model producing low-rated item recommendations.

Before implementing the SVD++ model, a transformation was applied to the play_count feature within the dataset. This transformation involved categorizing the play_count values into five distinct categories, ranging from 1 to 5. In this categorization scheme, higher category values indicate more positive user feedback or a higher level of user satisfaction with a specific song.

Table 3. Category description

Category	Description
1	Rarely (count < 3)
2	Occasionally (3 < count < 20)
3	Sometimes (20 < count < 100)
4	Usually (100 < count < 200)

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

Category	Description
5	Always (count >= 200)

Based on Table 3, the play_count values are classified into different categories based on specific thresholds. The rationale behind assigning these categories is to capture varying levels of user engagement and play counts for the songs. Values less than 3 represent songs with minimal user engagement and are categorized as 1. Values ranging from 3 to 19 are classified as category 2, which are songs that have received slightly higher user engagement compared to category 1 but are still considered to have relatively low play counts. Values ranging from 20 to 99 are assigned to category 3, representing songs that have a moderate level of user engagement, suggesting that they have received a significant number of play counts. Values between 100 and 199 are classified as category 4, which are songs that have achieved a relatively high level of user engagement, indicating the song is in the user's playlist. Values equal to or greater than 200 are designated as category 5. This category represents songs that have received a substantial amount of user engagement, indicating the user's favorite song.

Table 4. Preprocessed dataset sample

user_id	music_id	category
16c9d02c33e727…	SOYKKZP12A6D4F78D0	1
3ac4cc238575cd…	SOTGQIA12AF72A49C6	1
59340ae84bf17f…	SOBXHDL12A81C204C0	2

After performing data preprocessing, the dataset now contains user ID, music ID, and category of user's play count on the music as can be seen from a sample data in Table 4 above. Further on, the category feature will be used as the input for model training and recommendation generation later on.

3.2 Metric

The trained models are evaluated by calculating the accuracy of predictions using Root Mean Squared Error (RMSE). RMSE is measured on a scale of 0 to 1, where a value closer to 0 signifies a lower error and better performance. To calculate accuracy, the predicted values are compared to the actual values. RMSE is formulated as shown in equation 9.

$$RMSE = \sqrt{\frac{\sum_{(u,i)\epsilon TestSet}(\tilde{r}_{ui} - r_{ui})^2}{|TestSet|}} \quad (9)$$

where:

r_{ui} : original rating data of item i by user u
$\tilde{r}_{ui}$: predicted rating data of item i by user u
$TestSet$: testing dataset

In addition to RMSE, we use NDCG to evaluate the ranking of recommended items. The formula for NDCG can be seen in equation 10.

$$NDCG(u)@n = Z_u \sum_{j=1}^{n} \frac{G_{uj}}{log(1+j)} \quad (10)$$

where:

n : frequency of top items to be taken
Z_u : normalized parameter at rank j for user u
G_{uj} : gain of the item at rank j for user u

In this research, the evaluation using RMSE and NDCG will be conducted during the k-fold cross-validation process, and the results from each fold will be averaged. The frequency of items taken for NDCG is n = 5 and n = 10. To facilitate this, we utilize the scikit-learn library [20] for performing NDCG evaluation.

3.2 Experiment Result

Before conducting the cross-validation testing, a comprehensive exploration of the optimal values for the n_epochs and n_factors parameters were conducted. These parameters play a crucial role in the SVD++ model, influencing the number of iterations and the dimensionality of the latent factors used in the minimization process utilizing the stochastic gradient descent (SGD) algorithm.

To identify the optimal values, a series of tests were performed, encompassing various values for both n_epochs and n_factors. Specifically, we evaluated the values 10, 20, 50, 100, 150, and 200 for both parameters. During each test, assess the performance of the model by utilizing RMSE and NDCG@10. By conducting these tests, we sought to determine the combinations of n_epochs and n_factors that yielded the most favorable results in terms of accuracy and recommendation relevancy. This comprehensive exploration allowed us to identify the optimal parameter values that could enhance the performance of the SVD++ model and subsequently improve the accuracy and relevance of the recommendations generated.

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

Table 5. Experiment result on number of epochs

Epochs	Metric	
	RMSE	NDCG@10
n = 10	0.4437	0.8233
n = 20	0.4413	0.8233
n = 50	0.4431	0.8163
n = 100	0.4557	0.8049
n = 150	0.4638	0.8009
n = 200	0.4684	0.7986

The result of the experiment on number of epochs, as depicted in Table 5, shows a noteworthy correlation between the n_epochs parameter and the corresponding RMSE error values. Remarkably, an increase in the n_epochs value leads to a significant increase in the RMSE error, indicating a decline in the performance of the prediction model. However, when compared to subsequent n_epochs values, the error value results in a decrease when n = 20, implying a relatively improved model performance at this specific epoch value.

Similarly, when assessing the impact of the n_epochs value on item relevance with the NDCG@10 metric, a consistent trend emerges. The relevance of the recommendations appears to decline when the n_epochs value surpasses 50. Consequently, this finding suggests that higher epoch values may have a negative effect on the quality and relavancy of the recommended items.

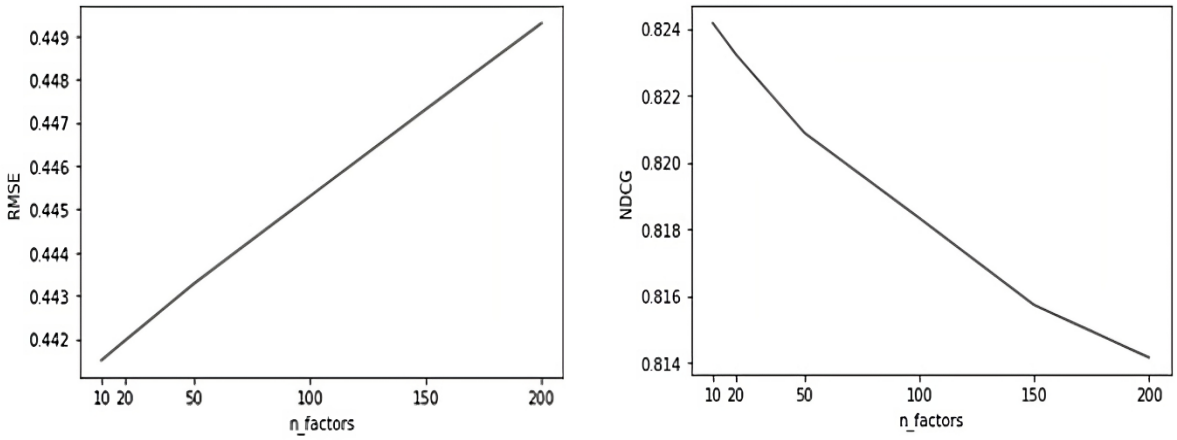

Figure 6. Experiment result on number of epochs

To further visualize the impact of the n_epochs value on the RMSE and NDCG@10 score discussed above, Figure 6 provides a graphical representation. This visualization allows for a more comprehensive understanding of how different epoch values impact the model's prediction accuracy.

Table 6. Experiment result on number of factors

Factors	Metric	
	RMSE	NDCG@10
n = 10	0.4415	0.8242
n = 20	0.4420	0.8232
n = 50	0.4433	0.8209
n = 100	0.4453	0.8183
n = 150	0.4473	0.8157
n = 200	0.4493	0.8141

The findings presented in Table 6 show an interesting pattern in relation to the n_factors parameter and its impact on the RMSE error values. Notably, a unique trend emerges, demonstrating an insignificant but consistent increase of 0.02 in the RMSE error when the n_factors value is incremented by multiples of 50. This pattern can be seen from the result for the values of n = 50, n = 100, n = 150, and n = 200, showcasing a linear relationship between n_factors and the corresponding RMSE error.

Furthermore, the relevancy score of NDCG@10 also shows a decline as the n_factors value increases. This observation implies that higher values of n_factors may negatively impact the relevance of the recommended items, potentially leading to less satisfactory user experiences.

To gain a more comprehensive understanding of the behavior and trends discussed above, Figure 7 provides a visual representation of the results.

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

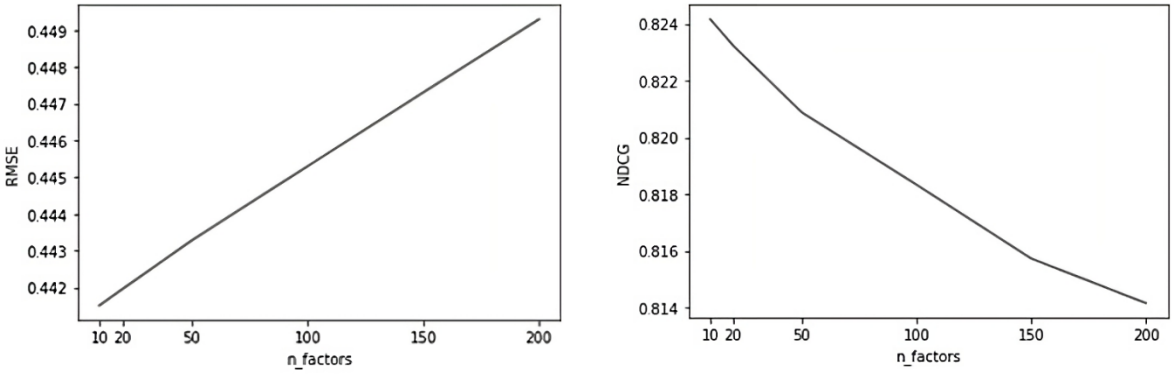

Figure 7. Experiment result on number of factors

Based on the experiment results discussed above, the RMSE value increases as the n_epochs value becomes less than or greater than 20. However, the NDCG show no significant difference between epoch values of 10 and 20, but the NDCG value continues to decrease when the epoch value surpasses 20, which indicates overfitting, indicating that the model may be excessively fine-tuned to the training data, thereby compromising its ability to generalize to new data. The n_factors parameter shows the same behavior as n_epochs with the best value of 10 as shown in Table 6 and Figure 7. Therefore, we will use an epoch value of 20 and factor value of 10 as the parameter for the SVD++ model.

Table 7. 5-fold cross-validation result

Iteration	Metric		
	RMSE	NDCG@5	NDCG@10
1	0.4413	0.8223	0.8235
2	0.4419	0.8231	0.8233
3	0.4430	0.8233	0.8231
4	0.4426	0.8234	0.8231
5	0.4427	0.8237	0.8231
Mean	0.4423	0.8232	0.8231

The cross-validation testing results for k = 5, as presented in Table 7, demonstrate that the RMSE scores obtained from each iteration show a relatively good value of accuracy in predicting user ratings. Additionally, the NDCG scores also indicate favorable outcomes. Notably, there are no significant differences in the metric scores across the iterations for this k-fold value.

Table 8. 10-fold cross-validation result

Iteration	Metric		
	RMSE	NDCG@5	NDCG@10
1	0.4417	0.7806	0.7803
2	0.4409	0.7798	0.7795
3	0.4414	0.7797	0.7797
4	0.4419	0.7792	0.7804
5	0.4405	0.7791	0.7801
6	0.4409	0.7799	0.7805
7	0.4414	0.7791	0.7802
8	0.4404	0.7807	0.7799
9	0.4411	0.7801	0.7795
10	0.4410	0.7796	0.7801
Mean	0.4411	0.7798	0.7800

Similarly, the cross-validation testing results for k = 10, shown in Table 8, shows that the RMSE scores obtained from each iteration show a relatively good accuracy in predicting user ratings. The NDCG scores also demonstrate satisfactory results. From the result, there are no significant differences in the metric scores for every iteration for this k-fold value.

However, when comparing the average values of NDCG@5 and NDCG@10 between k = 5 and k = 10, it is evident that the cross-validation with k = 10 yields slightly lower scores, with a difference of approximately 0.04. Despite this slight difference, the overall performance remains favorable for both k values. Considering the objective of minimizing the RMSE while maintaining a good recommendation relevancy, we have opted to select the model generated from the 5-fold cross-validation, specifically the model obtained in iteration 1.

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

Through this process, we have successfully determined the best model settings, specifically n_epochs = 20 and n_factors = 10, while utilizing a k-fold value of 5. These settings were chosen based on their ability to generate accurate predictions and relevant item recommendations, as evaluated through the cross-validation testing process. By leveraging these optimal model configurations, we expect an enhanced performance in the recommender system, leading to improved user experiences and more precise recommendations.

Table 9. Recommendation result

No.	Description	Rating
1	Working With Homesick	18.573
2	221	17.435
3	Clara meets Slope – Hard To Say	17.327
4	MIC (Speak Life Album Version)	17.212
5	Death Ain't No Big Deal (Tribute To Jake Hess Album Version)	17.140
6	Machine Kit	16.767
7	Cold Blooded (Acid Cleanse)	16.753
8	OVO MI JE `KOLA	16.462
9	Encore Break	16.462
10	In League With Satan	16.211

The item recommendations for the top 10 songs for a sample user are presented in Table 9. While the model's predictions are fairly accurate, the ratings assigned to the recommended songs do not align with this precision. In fact, the ratings are below 2, which is not a favorable value for a rating. We believe that this discrepancy is due to the non-normal distribution of the dataset used, which exhibits a significant disparity between low and high-value play counts. The dataset is skewed towards a higher prevalence of low-value counts.

Upon examining the recommendations for the top 10 songs generated for a sample user in Table 9, it becomes apparent that although the model's predictions are reasonably accurate, the assigned ratings for the recommended songs do not accurately reflect this level of precision. In fact, the ratings fall below 2, which is not an ideal value for items to be recommended. We propose that this inconsistency can be attributed to the non-normal distribution of the dataset utilized in the modeling process.

The dataset exhibits a substantial discrepancy between the low and high play counts, with a higher prevalence of low-value counts. This uneven distribution can potentially impact the modeling process, leading to lower ratings for the recommended songs. Consequently, despite the SVD++ model's ability to make relatively accurate predictions, the recommended items may not fully align with the user's true preferences for the songs.

Based on the evaluation results discussed above, despite performing k-fold cross-validation, the model built with deep learning approach utilizing Restricted Boltzmann Machine (RBM) in previous research conducted by Biswal et al. in 2020 [2] performed better compared to our SVD++ model trained with the same MSD The Echo Nest Subset dataset. Therefore, when building a recommender system for datasets with similar characteristics, employing a deep learning approach becomes a favorable option.

4. CONCLUSION

In this study, we developed a recommender system using collaborative filtering paradigm with the SVD++ algorithm. To construct our model, we utilized the MSD Subset The Echo Nest Dataset and considered user play count as implicit feedback and use it as the user rating for the music. In addition, we performed k-fold cross-validation to determine the most effective model for generating recommendations. From the experimentation results using the evaluation metrics RMSE and NDCG, we obtained the optimal value of 20 for n_epochs and 10 for n_factors for the SVD++ model. We conducted cross-validation with both 5 and 10 folds and observed that k = 5 yielded better performance, as indicated by an average RMSE of 0.4423, NDCG@5 of 0.8232, and NDCG@10 of 0.8231. It is noteworthy that the recommended items generated by our model tend to have low ratings. This observation can be attributed to the non-normal distribution of the dataset. Based on the findings of this research, it is recommended that future studies use datasets with a more normal distribution to achieve more accurate prediction results. Furthermore, future studies could explore alternative collaborative filtering techniques within the same research context to facilitate comparisons. This approach would enable meaningful comparisons and provide additional insights into the effectiveness and efficiency of different methods.

REFERENCES

[1] M. Schulz, S. Weinzierl, and J. Steffens, "The Impact of Choice Overload on Music Listening Experience," 2021.
[2] A. Biswal, M. D. Borah, and Z. Hussain, "Music recommender system using restricted Boltzmann machine with implicit feedback," in Advances in Computers, Academic Press Inc., 2021, pp. 367–402. doi: 10.1016/bs.adcom.2021.01.001.
[3] Z. Romadhon, E. Sediyono, and C. E. Widodo, "Various Implementation of Collaborative Filtering-Based Approach on Recommendation Systems using Similarity," Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, pp. 179–186, Jul. 2020, doi: 10.22219/kinetik.v5i3.1062.

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 7, Nomor 3, Juli 2023, Page 1210-1219
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI: 10.30865/mib.v7i3.6424

[4] M. Jalili, S. Ahmadian, M. Izadi, P. Moradi, and M. Salehi, "Evaluating Collaborative Filtering Recommender Algorithms: A Survey," IEEE Access, vol. 6, pp. 74003–74024, 2018, doi: 10.1109/ACCESS.2018.2883742.
[5] A. M. A. Al-Sabaawi, H. Karacan, and Y. E. Yenice, "Svd++ and clustering approaches to alleviating the cold-start problem for recommendation systems," International Journal of Innovative Computing, Information and Control, vol. 17, no. 2, pp. 383–396, 2021, doi: 10.24507/ijicic.17.02.383.
[6] M. Srifi, A. Oussous, A. A. Lahcen, and S. Mouline, "Recommender systems based on collaborative filtering using review texts-A survey," Information (Switzerland), vol. 11, no. 6. MDPI AG, Jun. 01, 2020. doi: 10.3390/INFO11060317.
[7] D. Jannach, L. Lerche, and M. Zanker, "Recommending based on implicit feedback," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer Verlag, 2018, pp. 510–569. doi: 10.1007/978-3-319-90092-6_14.
[8] Q. Zhao, F. M. Harper, G. Adomavicius, and J. A. Konstan, "Explicit or implicit feedback? engagement or satisfaction?: A field experiment on machine-learning-based recommender systems," in Proceedings of the ACM Symposium on Applied Computing, Association for Computing Machinery, Apr. 2018, pp. 1331–1340. doi: 10.1145/3167132.3167275.
[9] D. Bokde, S. Girase, and D. Mukhopadhyay, "Matrix Factorization model in Collaborative Filtering algorithms: A survey," in Procedia Computer Science, Elsevier B.V., 2015, pp. 136–146. doi: 10.1016/j.procs.2015.04.237.
[10] Rachana Mehta and Keyur Rana, "A review on matrix factorization techniques in recommender systems," in Proc. 2nd Int. Conf. Commun. Syst. Comput. IT Appl. (CSCITA), 2017, pp. 269–274. doi: 10.1109/CSCITA.2017.8066567.
[11] S. Jiang, J. Li, and W. Zhou, "AN APPLICATION OF SVD++ METHOD IN COLLABORATIVE FILTERING," in 2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), 2020, pp. 192–197. doi: 10.1109/ICCWAMTIP51612.2020.9317347/20/$31.00.
[12] H. Tian, H. Cai, J. Wen, S. Li, and Y. Li, A Music Recommendation System Based on logistic regression and eXtreme Gradient Boosting. 2019. doi: 10.1109/IJCNN.2019.8852094.
[13] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, "The Million Song Dataset," in Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR), 2011. doi: 10.7916/D8377K1H.
[14] Y. Dong, X. Guo, and Y. Gu, "Music Recommendation System based on Fusion Deep Learning Models," in Journal of Physics: Conference Series, Institute of Physics Publishing, Jun. 2020. doi: 10.1088/1742-6596/1544/1/012029.
[15] G. Liu and X. Zhao, "Recommender System for Books in University Library with Implicit Data," in Proceedings of the 2018 International Conference on Network, Communication, Computer Engineering (NCCE), 2018, pp. 164–168. doi: 10.2991/ncce-18.2018.28.
[16] A. Pujahari and D. S. Sisodia, "Model-Based Collaborative Filtering for Recommender Systems: An Empirical Survey," in 2020 First International Conference on Power, Control and Computing Technologies (ICPC2T), 2020. doi: https://doi.org/10.1109/ICPC2T48082.2020.9071454.
[17] S. H. Chen, S. I. Sou, and H. P. Hsieh, "HPCF: Hybrid Music Group Recommendation System based on Item Popularity and Collaborative Filtering," in Proceedings - 2020 International Computer Symposium, ICS 2020, Institute of Electrical and Electronics Engineers Inc., Dec. 2020, pp. 43–49. doi: 10.1109/ICS51289.2020.00019.
[18] J. Jiao, X. Zhang, F. Li, and Y. Wang, "A Novel Learning Rate Function and Its Application on the SVD++ Recommendation Algorithm," IEEE Access, vol. 8, pp. 14112–14122, 2020, doi: 10.1109/ACCESS.2019.2960523.
[19] N. Hug, "Surprise: A Python library for recommender systems," Journal of Open Source Software, vol. 5, no. 52, p. 2174, Aug. 2020, doi: 10.21105/joss.02174.
[20] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011, doi: 10.48550/arXiv.1201.0490.

Submitted: **21/06/2023**; Accepted: **17/07/2023**; Published: **23/07/2023**