CSE258 Project Report:
Amazon Digital Music Recommendation

1 Introduction

Recommender systems are utilized in a variety of areas and are most commonly recognized as playlist generators for video and music services like Netflix, YouTube and Spotify, product recommenders for services such as Amazon, or content recommenders for social media platforms such as Facebook and Twitter. These systems can operate using a single input, like music, or multiple inputs within and across platforms like news, books, and search queries. There are also popular recommender systems for specific topics like restaurants and online dating. Recommender systems have also been developed to explore research articles and experts, collaborators, and financial services [4].

The recommender system deals with a large volume of information present by filtering the most important information based on the data provided by a user and other factors that take care of the user's preference and interest. It finds out the match between user and item and imputes the similarities between users and items for recommendation. [10].

In this project, we are building a digital music recommendation system based on Amazon product review. We utilize several trending methods like Popularity-based filtering, text mining, semantic analysis.

The report is structured as follows: in the second section, we describe in detail the dataset we are using to make predictions and recommendations. In the third section, we introduce the literature related to the problem we are studying, and compare other datasets similar to ours. In the fourth section, we describe the specific task in our project and the motivation behind it. In the fifth section, we elaborate the four specific models we choose to complete the prediction and recommendation task. In the sixth section, we show our experimental results and analysis. In the seventh section, we give our conclusions. And in the last section, we discuss other literature similar to our work.

2 Dataset Description

In this project, we are using the Digital Music dataset [7]. The columns of our dataset includes:

- overall - the rating of the product
- vote - helpful votes of the review
- verified - a boolean value indicating if the review has been verified
- reviewTime - time of the review (raw)
- reviewerID - ID of the reviewer
- asin - ID of the product
- style - a disctionary of the product metadata
- reviewerName - name of the reviewer
- reviewText - text of the review
- summary - summary of the review
- unixReviewTime - time of the review (unix time)

And here are some basic statistics and properties of our dataset:

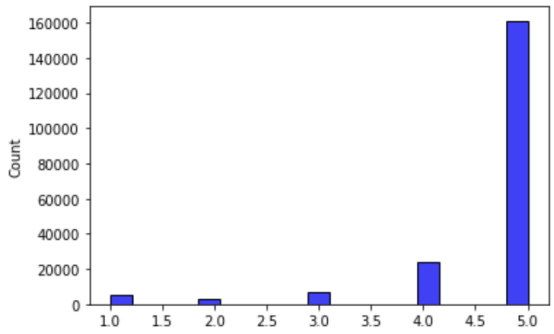
- There are 1584082 pieces of data in total.
- There are 1582629 pieces of data which has the 'reviewText' column, which is about 99.91% of all data.
- There are 1216329 pieces of data which has the 'reviewText' column and is verified, which is about 76.78% of all data.

In this project, we are going to build our model based on the first 200,000 data. And we can see, among these data:

- There are 199901 pieces of data which has the 'reviewText' column, which is about 99.95% of all data.
- There are 144397 pieces of data which has the 'reviewText' column and is verified, which is about 76.85% of all data.

We observe that the percentages of the first 200,000 data are really close to the percentages of the whole dataset. Therefore, by building models utilizing the first 200,000 data, the conclusion should also apply to the whole dataset.

The figure below shows the rating score distribution across the entire data set.



Rating distribution

3 Related Work

In this section, we mainly discuss some common recommender systems that are broadly used in industries and academic research. Recommender systems are broadly classified into three categories: collaborative filtering (CF), contents-based methods, and hybrid methods [8].

Collaborative filtering uses only user-item rating matrix for predicting unseen preference. The most effective memory-based algorithms known so far is item-based CF [2]. Collaborative filtering systems recommend items based on similarity measures between users and/or items. The items recommended to a user are those preferred by similar users. This sort of recommendation system can use the groundwork laid on similarity search and on clustering.

Content-based systems examine properties of the items recommended. For instance, if a Netflix user has watched many cowboy movies, then recommend a movie classified in the database as having the "cowboy" genre. Pazzani[3] studied this approach in depth, including how to build user and item profiles.

Hybrid approach combines both collaborative and content-based recommendation. Koren[9] suggested effectively combining rating information and user, item profiles for more accurate recommendation.

4 Tasks Description

Item prediction(binary) and rating prediction have always been topics that attract people's attention. These predictions could be used in order to:

1. Personalize recommendation
2. Detect suspicious or fake online reviews
3. Better analyze and understand customers

In the following tasks, we are using accuracy(for Item prediction) and mean square error (for rating prediction) as evaluation methods for our models.

5 Model Description

In this section, we will describe our five recommendation models.

5.1 Popularity-based Filtering

The first model we use is popularity-based filtering model. Popularity-based recommendation system

works with the trend. It basically uses the items which are in trend right now. For example, if any music which is usually heard by every new user then there are chances that it may suggest that item to the user who just signed up. This model serves as a baseline for our recommendation system.

The popularity-based filtering model works as follows: the model returns either "recommend" or "not recommend". For simplicity, we denote them as "YES" and "NO" correspondingly. The model will return "YES" if the designated piece of music is considered *popular*. By *popular* we mean that it accounts for a certain threshold τ (discussed later in section six) of the total user- music interactions.

$$I_{popular} = \begin{cases} 1, & \frac{count}{totalHeard} > \tau \\ 0, & otherwise. \end{cases}$$

where $I_{popular} = 0$ means "NO" and $I_{popular} = 1$ means "YES"; and *count* denotes the number of times that music has been heard by users, and *totalHeard* denotes the total number of times of all user-music interactions.

5.2 Regression

Regression model is one of the widely used models in machine learning. In this section, we are going to build several regression models and compare them.

First, we are going to train a simple predictor that estimates the 'overall'(which is the rating) from the number of times the exclamation mark (!) symbol is used in the reviewText. We utilized the numpy function: numpy.linalg.lstsq to calculate the theta in our model and report the mse(mean squared error).

Second, we re-train the predictor so as to include a second feature based on the length of the reviewText. Again, we utilized the numpy function: numpy.linalg.lstsq to calculate the theta in our model and report the mse(mean squared error).

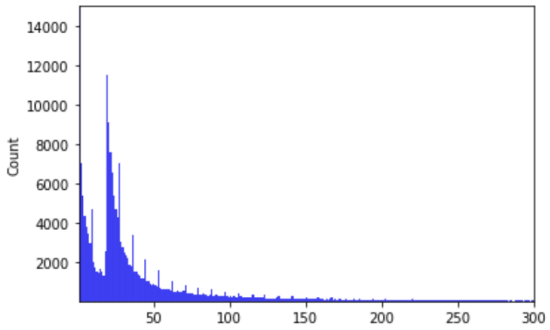
Third, we train a model that fits a polynomial function to estimate ratings based on the number of times the exclamation mark (!) symbol is used in the reviewText and report the MSE.

For the three models above, we are going to build each of them twice, one using the data with 'reviewText', and the other one using the data with 'reviewText' and 'verified' is TRUE.

5.3 Sentiment Analysis

For this approach, we predict user's rating based on their review text. We are using a sentiment analysis model that predicts ratings from a N word bag-of-words model (based on the most popular words). Using the 180,000 first reviews for training and the rest for testing, the review texts are processed to be without capitalization or punctuation. The number of unique words in the review text data: 222619

The figure below shows the review text length distribution across the entire data set.



Review Length Distribution

5.4 Rating Prediction with Latent factor

In the recommender system, rating prediction is an important task to quantify the user's preference. In this section, we choose the simple latent factor model to predict the user's preference. It consist the following parameters:

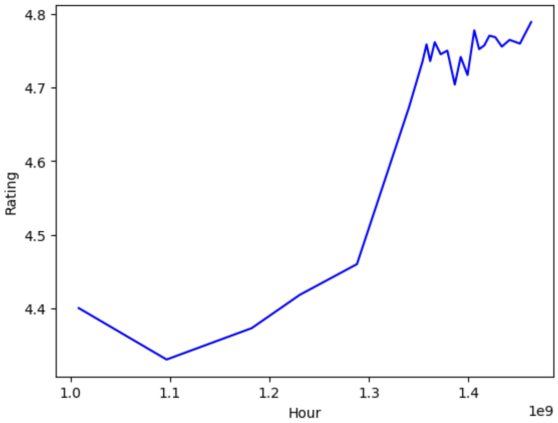
$$f(u,i) = \alpha + \beta_u + \beta_i + \lambda \left[\Sigma_u \beta_u^2 + \Sigma_i \beta_i^2\right]$$

Which f(u,i) represents the predicted rating, u represents a user, i represents a music item. Alpha represents the rating mean, Beta represents bias term, Lambda is the regularizer. This model uses gradient descent to find the *argmax* of the user bias term and item bias term from user/item's past rating. With this model, it can capture the rating dynamic by incooperating user and item specific feature that affects rating.

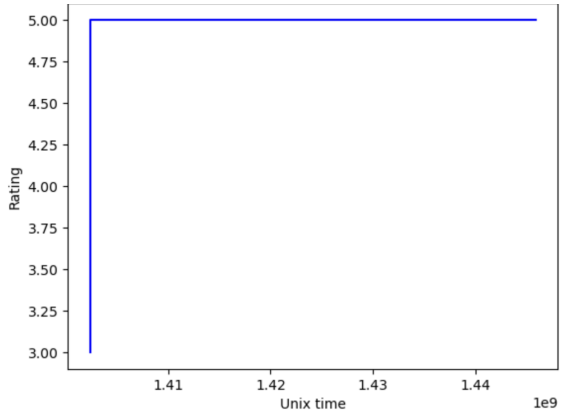
5.5 Temporal Analysis with Rating Prediction

In the recommender system, temporal data is very important for rating prediction. Seasonal and peri-

odic change can affect people's preference in music. It is also pretty common to have the rating on an item increase in time because of nostalgic reason. To explore how much temporal factor influenced the user rating. We graph out the rating over time for a random item, a random user. And with the sliding window technique, we graph out the rating from training dataset across the time(each point in the graph represents the average rating in a 2 hours window size).



Ratings over Time (Window Size = 2 hours)

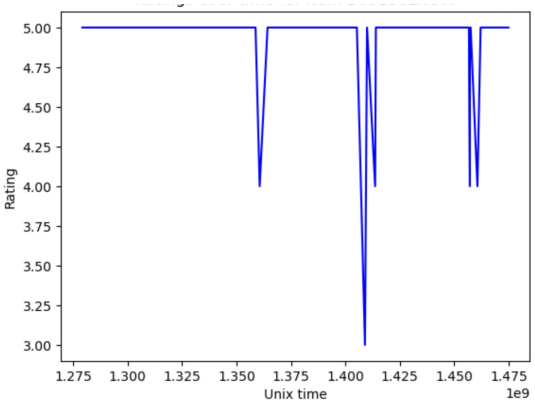


Ratings over Time for User A1MAZQUR7WF4

From the figures analysis above, there are a couple rating trends incorporate with temporal feature. As a result, we build a simple regression model with different time windows(last twelve hours, one day ago, one week ago, one month ago, three months ago) and a intercept term as features for each data point to predict rating, and a model incorporate latent factor rating prediction in section 5.4 with a time bias term, which writes as:

$$f(u,i) = \alpha + \beta_u + \beta_i + \beta_t + \lambda \left[\Sigma_u \beta_u^2 + \Sigma_i \beta_i^2\right]$$

Which Beta t is calculated by average of rating over a time period t times a regularizer term, lambda t.

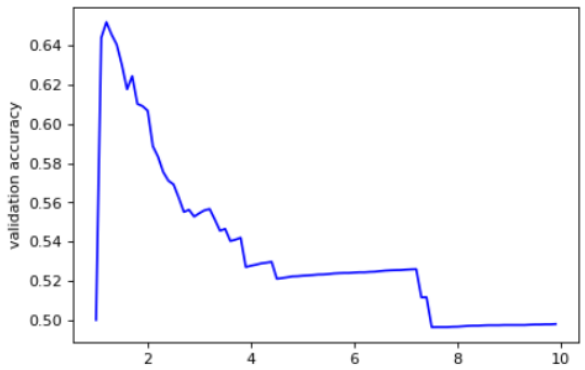


Ratings over Time for Item B00136LNOW

6 Experimental Results

6.1 Popularity-based Filtering

As discussed in 5.1, we explored a series of thresholds to determine the popularity of each music. We searched from $\tau = 1$ to $\tau = 10$, with a step size of 0.1. The recommendation accuracy trend is as shown in Figure. 2. From the curve we can derive that the best threshold is 1.20, and the corresponding accuracy is 0.65195.



The relationship between Validation Accuracy and Popularity Threshold

6.2 Regression

As discussed in 5.2, we first observe our first regression model that estimates the 'overall'(which is the rating) from the number of times the exclamation mark (!) symbol is used in the reviewText. We achieved the mse of 0.710 using the data with reviewText. And we achieved the mse of 0.617 using the data with reviewText and is verified.

Second, we observe our second regression model that estimates the 'overall'(which is the rating) from the number of times the exclamation mark (!) symbol is used in the reviewText and a second feature based on the length of the reviewText. We achieved the mse of 0.699 using the data with reviewText. And we achieved the mse of 0.552 using the data with reviewText and is verified.

Finally, we observe our third regression model which fits a polynomial function to estimate ratings based on the number of times the exclamation mark (!) symbol is used in the reviewText. We achieved the mse of 0.707 using the data with reviewText. And we achieved the mse of 0.556 using the data with reviewText and is verified.

By observing our models and the corresponding output, we achieve the following conclusions:

1. The model utilizing verified data generates better accuracy. By oberving the three models, we find out that in every situation, the model used verified data always has the smaller mse than the other model. Thus, we can achieve our conclusion that the model utilizing verified data generates better accuracy.

2. The length of the reviewText contributes to the prediction of rating. By observing model one and model two, we notice that the second model which estimates the rating from the number of times the exclamation mark (!) symbol used in the reviewText and a second feature based on the length of the reviewText performs better than the first model which estimates the rating only from the number of times the exclamation mark (!) symbol used in the reviewText; no matter using the verified data or not. Thus, we conclude that the length of the reviewText contributes to the prediction of rating.

3. The model which estimates the rating from the number of times the exclamation mark (!) symbol used in the reviewText and a second feature based on the length of the reviewText performs the best among the three models. We notice that the second model has the lowest mse, no matter which dataset we are using.

6.3 Sentiment Analysis

As mentioned 5.3, we are using a sentiment analysis model that predict ratings from a N word bag-of-words model (based on the most popular words). A Ridge regression model is used for prediction. As N is increased from 1,000 to 3,000, the Mean Squared Error of the unigram model has been improved from 0.909 to 0.859. As N is increased from 1,000 to 5,000, the Mean Squared Error of the bigram model has been improved from 0.971 to 0.890.

6.4 Rating Prediction with Latent factor

As mentioned in 5.4, we are using a latent factor model that predict ratings. With lambda range from 10^{-6} to 10, the Mean Squared Error of minimized at lambda = 10^{-5} , which has mean square error = 1.368 for rating prediction on test set.

6.5 Temporal Analysis with Rating Prediction

As mentioned in 5.5, we first used regression to build a linear regression model with different time windows(last twelve hours, one day ago, one week

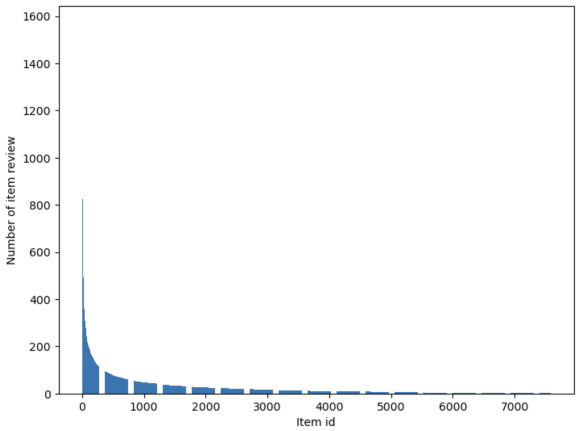
ago, one month ago, three months ago) and a intercept term as features for each data point to predict rating. The theta values are -7.46166946e+09, 7.22408295e-04, 1.02604151e-01, 2.56383419e-01, 5.60434103e-01, 3.49074308e+10. From the data we can see, the rating one month ago contributes the most to predict current rating. The mean square error of the model is 1.393 compares to the baseline model's 1.411, which predicts the mean rating all the time.

Second, we used a latent factor model with time bias parameter to predict rating. With time window t = 672 hours (one month), and regularizer lambda =1, we are able to reduce the mean square error of the latent factor model without time bias from 1.368 to 1.359.

7 Conclusions

From the five models we built, we have following conclusions:

For our item prediction model(popularity-based Filtering), we have an accuracy level of 0.65. This model has a particular advantage for new user, who does not have previous history data to be leaned from. But there is also room for improvement for this dataset. From graph below, the majority of the item has similar amount of review, which is less than 100. In the case of item is music track, a heuristic can be used to improve this model is to use item similarity to do item prediction. If this music track does not appear in the popularity set, we can reconsider recommending by checking the item's similarity to the other items the current user likes/dislikes. For example, if a user previously had liked the music from the same artist, they have a big chance wanted to be recommended the music from the same artist again.



The Item Review Distribution

From our rating prediction models, the regression model with the heuristic of using the length of the review text and the count of exclamation mark has the lowest mean square error. From the rating distribution of the dataset, it is explainable in the way that the dataset is imbalanced towards the higher rating end. The heuristic, successful used the features that closely related to high rating review semantic. However, there is also drawback with this model if the dataset is more neutral on the rating, or lack of review text data.

Likewise, the model with semantic analysis can also face the challenge of lacking review text data, but it can handle neutral rating semantic and adapt in culture settings on language habit better. But there is also the consideration of cost of the ngrams modeling, which in some case can be really expensive.

Third, the latent factor model can be not very predictive in strong individualized product setting. In this case, a user's music track's rating has a relatively weak correlation with how other people's rating of the music, compares the items that have less dimensions to consider in rating (such as items mostly being rated for quality and duality). An improve of the model, likewise mentioned as improvement for popularity model, is to expand the latent factors being measured in the model and factor in user/item interaction instead of viewing user bias and item bias as two separated terms.

Last, the exploration of temporal factor brought some intuitions for the last two models. From the graph of rating change over time, we can observe there is some trend with rating across time. And

it can be reassured from the first temporal model with simple linear regression. All the data being used are just timestamp and previous ratings, which is relatively cheap, but it still outperforms the baseline model. Another more effect way to make use of temporal data is to incorporate it with models with other features. In this case we choose the latent factor model to be incorporated, and it showed a good amount of improvement. Same thing can be done with other models such as factorized machine(FM) model and Factorizing Personalized Markov Chains(FPMC) model.

8 Literature Reivew

In this project, we utilized the Amazon Review Data(2018) and performed several models on the dataset. Review datasets are widely used in recommendation system studying. Since item and rating predictions have always been among the top topics of recommendation system. And in this project, we employed Digital Music of Amazon review data(2018), which is the review and product metadata for the digital music category. And Amazon, as one of the most popular e-commerce companies nowadays, provides us with lots of dataset that can be studied.

We find out that the Amazon Review Datasets have been studied by a lot of scholars. For example, some scholars from Stanford University have performed sentiment analysis for Amazon Reviews. Algorithms including Naive Bayes, SVM(Support Vector Machine), KNN(K-nearest Neighbors Algorithm) and LSTM(Long Short-Term Memory) have been used to study the dataset [11].

The current sentiment analysis model is not comparable to other models, since it is only predicting ratings on items that users have written review of. For future improvements of the sentiment analysis, we can extract user preference from their reviews and comments to suggest items according to their preferences. Specifically, analyze their sentiments towards the songs and recommend similar songs to them based on similarity to their preference. [5]

While we explored the use of temporal dynamic in recommender system, the paper Vista: A paper Visually, Socially, and Temporally-aware Model for Artistic Recommendation in 2016 showed the models used in industry to give individualized recommendation. In the paper, a higher order personalized Markov chain algorithm is used to model the sequential and temporal data, which incorporate the interaction between user and item (long-term dynamics) tl, and interaction between item and user (short-term dynamics) ts, as well as a decaying scheme applied to capture the interaction weight overtime(rather than assume independent action). It is a relatively light weight model and has a lot of scalability in the industry setting.[6]

As for popularity-based filtering, it is a common solution for the user-cold start problem. When there is insufficient information about the interests and preferences of a new user or visitor in the system, the system can predict items that suit the user's interest by recommending popular items to users.

In most recent research, H. J. Ahn implemented popularity in his hybrid movie recommender system by combining information based on popularity-class and genre [1]. His work introduces a three-dimensional popularity concept including average rating, percentage of being rated, and strong support as an approach to discover a buyer's preferred popularity characteristics and provide recommendations based on these characteristics, which is more thorough than the model we used in our experiment.

References

[1] AHN., H. J. Utilizing Popularity Characteristics for Product Recommendation, International Journal of Electronic Commerce, 2006.

[2] BASED COLLABORATIVE FILTERING RECOMMENDATION ALGORITHMS.., I. *B. Sarwar, G. Karypis, J. Konstan*. 2001.

[3] BASED RECOMMENDATION SYSTEMS, C. *M. Pazzani and D. Billsus.* 2007.

[4] DWIVEDIAPR, R. What Are Recommendation Systems in Machine Learning?

[5] ELHAM ASANI, HAMED VAHDAT-NEJAD, J. S. Restaurant recommender system based on sentiment analysis.

[6] HE, R. Vista: A Visually, Socially, and Temporally-aware Model for Artistic Recommendation.

[7] JIANMO NI, JIACHENG LI, J. M. *Justifying recommendations using distantly-labeled reviews and fined-grained aspects*. 2019.

[8] JOONSEOK LEE, K. L. *Personalized Academic Research Paper Recommendation System, 4th.Edition*. 2017.

[9] MEETS THE NEIGHBORHOOD: A MULTIFACETED COLLABORATIVE FILTERING MODEL., F. *Y. Koren*. 2008.

[10] SEN, S. Some major recommender system algorithms.

[11] WANLIANG TAN, XINYU WANG, X. X. *Sentiment Analysis for Amazon Reviews*. 2018.