AAI627

Team SLG: Jichen Li, Yipeng Sun, Linfeng Ge

December 18, 2022

1 Introduction

Music Recommendation System whether a user will like a particular song or not. The training data includes A user's rating history for tracks, albums, artists and genres. A track's track is on an album. An album is with an artist. An artist is associated with one or more genres. The user can rate any item with a score between 0 and 100. The task is to find three tracks out of six given tracks that the user might like. We used multiple algorithm to do this project.

2 Methods used in experiment

2.1 Linear Regression

The linear regression prediction method is to find the causal relationship between variables, express this relationship with a mathematical model, and calculate the correlation degree of these two variables through historical data, so as to predict the future situation. By collating the scores in the training file and the corresponding author albums in the prediction file, we can get a list of corresponding scores. Afterwards, we get the final score by setting the ratio of author and album scores, and use this as the most basis to get the top three and bottom three, which are the prediction results.

First we try to calculate the score with the following equation:

$$Score_i = 0.4 * album_i + 0.3 * artist_i + 0.3 * genre_i$$

and the accuracy was 0.78113. After adjusting the parameter to:

$$Score_i = 0.7 * album_i + 0.2 * artist_i + 0.1 * genre_i$$

the accuracy increased to 0.87154. We also tried other combinations and the accuracy changed between 0.86 to 0.87.The interesting thing is that the accuracy is increased when we increase the weight of the album score from 0.4 to 0.7 and decrease the weight of the genre score from 0.3 to 0.1. It suggests that people are more likely to like music from the same album than from the same genre.

The conclusion is clear after several experiments. When evaluating a music track, the album score is more important than the artist score, and far more important than the genre score.

2.2 Matrix Factorization

Matrix factorization is a way to generate latent features when multiplying two different kinds of entities. It can represent users and items in two lower dimensional latent spaces, the first one has a row for each user, while the second has a column for each item. The row or column associated to a specific user or item is referred to as latent factors. The predicted ratings can be computed as $\tilde{R} = HW$, where $\tilde{R} \in R^{users \times items}$ is the user-item rating matrix, $H \in R^{users \times latentfactors}$ contains the user's latent factors and $W \in R^{latentfactors \times items}$ the item's latent factors. The goal is to minimize the following objective function:

$$\arg\min_{H,W} \|R - \tilde{R}\|_{\mathrm{F}} + \alpha\|H\| + \beta\|W\|$$

There is an algorithm in pyspark called Alternating Least Squares (ALS) matrix factorization which attempts to estimate the ratings matrix R as the product of two lower-rank matrices. First we set

the ALS parameter as $rank = 10$ and the accuracy is 0.54879. Then we changed to $rank = 20$ and the accuracy is 0.5907. I think the result will be better if we increase the maxIter and rank size. Howeverwhen the parameters increase, the computational effort increases accordingly. And it's hard to try larger parameter on Colab.

2.3 Logistic Regression

Pyspark.ml is a package in the PySpark library that provides a set of machine learning algorithms for building and tuning models. It is designed to be used in conjunction with the PySpark library, which is a Python library for distributed data processing that is built on top of the Spark framework. pyspark.ml provides a number of useful tools for working with machine learning models in PySpark, including:

- A suite of machine learning algorithms for classification, regression, clustering, and collaborative filtering.
- Functions for preprocessing and manipulating data, including one-hot encoding, standardization, and feature extraction.
- Tools for model selection and evaluation, including cross-validation and hyperparameter tuning.

To use pyspark.ml, you will need to have PySpark and the required dependencies installed. You can then import the necessary classes and functions from the pyspark.ml package and use them to build and evaluate machine learning models. We were given an file on Canvas. It has training data in format of userId, itemId and score. This data is more conducive to supervised machine learning. We try to use PySpark to deal with this problem. We are import PySpark machine learning library which is help people processing big data by machine learning ways. So we try to use this method to handle our final project's data. At first we need to pre-process the data using the tools in PySpak. Remove features that are not useful for the final model training and keep only the features we need. Then divide the training data and test data. Ensure that the data is correctly formatted. secondly,We use PySpark to process the data and put the target data into our already trained model in the format of the train data.Finally, we used the trained model to predict our target data and finally obtained good prediction results.

2.4 Decisoion Tree

A decision tree is a tree structure, either binary or non-binary, or it can be considered as a collection of if-else rules, or as a conditional probability distribution over a feature space. The internal node in the tree represents an attribute, the branches leading from the node represent all possible values of this attribute, and the leaf node represents the final classification result. One rule is constructed for each path from the root node to the leaf nodes, and these rules are mutually exclusive and complete in the sense that each sample is covered by one and only one path.

Thus, we can also use decision tree to do this work. Decision trees are also good classification models. The workflow of everything before modeling is the same as before. However, now we try use sklearn to do this. So we will use sklearn to do all work, include preprocessing data and training model. Here is the decision tree model that we have trained. We use this model to make the final prediction.

2.5 Gradient Boosting Decision Tree

The decision tree used by GBDT is a CART regression tree. Whether it is dealing with regression problems or binary classification and multi-classification, the decision trees used by GBDT are all CART regression trees. Each iteration of GBDT needs to fit the gradient value, which is a continuous value, so a regression tree is used. The criterion for the best division point in the classification tree is entropy or Gini coefficient, which is measured by purity, but the sample label in the regression tree is a continuous value, so it is no longer appropriate to use indicators such as entropy and replace it. is the squared error, which is a good measure of the fit.

2.6 Random Forest Classifier

Random forest is a very representative Bagging ensemble algorithm. All its base evaluators are decision trees. The forest composed of classification trees is called random forest classifier, and the forest integrated with regression trees is called random forest Classifier.

The key steps in the random forest construction process (M represents the number of features):

- Randomly select a sample at a time, with replacement sampling, repeat N times (repeated samples may appear)
- Randomly select m features, m $\leq$ M, and build a decision tree

3 Ensembling Method and results

3.1 Ensemble method

At the end, we experimented with the ensembling method. We can construct an optimized linear combination of all predictions to approximate the ground truth solution. We use the information in the previously submitted prediction files to make them form a large matrix, and then use the LS solution to complete the minimum distance optimization problem. Finally, a new prediction matrix is completed through matrix conversion and calculation. The function is as follow:

$$s_{ensemble} = a_1 s_1 + a_2 s_2 + \cdots + a_k s_k = S(S^T S)^{-1} S^T x$$

and $S^T x$ has been demonstrated to be reformed as:

$$S^T x = \begin{bmatrix} N(2P_1 - 1) \\ N(2P_2 - 1) \\ \vdots \\ N(2P_K - 1) \end{bmatrix}$$

We have ensembled nearly 40 results and this is all we have attempted to do.

Item	Best Score
Linear Regression	87.154%
Matrix Factorization	59.070%
Decision Tree	85.283%
Gradient-Boosted Tree Classifier	86.658%
Random Forest Classifier	87.017%
Logistic Regression	86.731%
Ensembling Method	87.346%

Table 1: Methods and their scores

3.2 Result

All the methods and corresponding scores we used are documented in the table above, and it can be seen that the ensemble score is the best among them all. The results of the other methods we used were not bad but not satisfying to the whole team. After analyzing all methods and discussions, we think that some parameters should be changed in the algorithm with better scores and ensemble, such as modifying iteration parameters and learning rate for more tests. In our tests, such as the logistic regression algorithm, we improved the score from 84.44% to 86.73% by tweaking the details.

3.3 ItemCF

We also tried to apply Item-based collaborative filtering(ItemCF) with pyspark. ItemCF is the recommendation system to use the similarity between items using the ratings by users. The function to

calculate the similarity between item A and item B is as follow:

$$similar_{AB} = \frac{|N(A) \cap N(B)|}{\sqrt{|N(A)|\,|N(B)|}}$$

where $N(A)$ represent the number of users who like item A and $N(B)$ represent the number of users who like item B. We have calculated the item similarity matrix and can recommend items to one user. However it's difficult to recommend for all 20000 users. Colab cannot support such a large amount of computation. We will attach the ItemCF code but there is no recommend result for Kaggle.

4 Conclusion

The above is what we tried in the project. Each method got different prediction scores. Some scores were high and some were low, and finally we integrated these results and predicted the output through the integration algorithm to get the best prediction score.