Practical 3: Recommendation System

Team 27: Gemini921
Xinyuan Wang wang_xinyuan@g.harvard.edu
Rui Zhao ruz466@g.harvard.edu
Yijun Zhou yijun_zhou@g.harvard.edu

1 Technical Approach:

1.1 Data Exploration & Feature Engineering:

- Extract Artists' Features:

  From the provided files, we got the profiles for users without artists' information, so we decided to extract all the artists data from the musicbrainz API via its python package. We extracted artists' "name", "type", "country", "life-begin", "life-end" and "begin-area-name", for we believed these features are important for the preference of users towards different artists. Meanwhile, we already have "sex", "age" and "country" as the users' features. Thus we combined the users' and artists' information together on the training data for feature engineering.

- Feature Engineering:

  In order to prevent collinearity and excessive dimension expansion, we first deleted some features of the artists including "life-end" and "begin-area-name", for most of the life-end information is none, and the information of begin-area-name is mostly contained in "country". Secondly, since many features of users and artists are categorical, we applied one-hot encoding to convert them to numerical for model fitting.

  Now we got 224 numerical features in total. We initially considered using PCA to reduce the dimension of feature space; however, it is actually unnecessary since 224 is a reasonable size compared with the size of training data. Low feature dimension with considerable training dataset may lead to lower prediction accuracy. Therefore, we decided not to use PCA for dimensionality reduction or other matrix factorization techniques.

1.2 Model Selection:

- Regression:

  After the feature engineering, we got a 4154804*224 feature matrix with all the numerical users' and artists' features. With the labels (play counts) in train data, we first fitted some regression models: **Lasso Regression**, **Ridge Regression**, **Decision Tree**(segmenting

the predictor space and use the mean in the segment of the response variable), **Random Forest**(building a number of decision trees on bootstrapped training samples), and **Ensemble Method**(an ensemble of comparatively weak prediction models).

However, it turned out that the training data is too large for regression, so we decided to take a subset to compare the performance of the models. We selected the first 10000 users from profile.csv, and took all the entries containing the selected users out of train.csv to make sure we trained on all known information. The MAE scores are listed in Table 1; none of them beat the baseline model. Due to the computationally expensive situation and the poor performance on the sample training set, we decided not to apply the regression methods here.

- K-means user clustering:

  K-means clustering is a method of vector quantization that is popular for cluster analysis in data mining. K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. Here since we have too many training data (user-artist pairs), we would classify users into different clusters first and then make predictions for new observations. In this way, we can speed up the process. We clustered the users into k clusters using users' features. For each specific artist, all the users in the same cluster share the same play counts, i.e. the median of all the users' play counts with respect to that artist. In this way, we took both user and artist into account and reduced the computational complexity.

  With K-means we were able to get results on the entire train set. We used 5-Fold cross validation and examined K from 1 to 600 for every 50 counts. From the graph below we could see that K = 50 got the best result. This means smaller K functioned better.

- Multinomial logit models:

  By carefully examining the data, we decided to experiment with the Multinomial Logit model. Every user with its features is treated as an observation, and can be seen as having a multinomial distribution over the 2000 artists. For example, for the i-th user, treating the row totals as given, counts in the K = 2000 categories are multinomially-distributed with

  $$(Y_{i1}, Y_{i2}, ..., Y_{i1999}, Y_{i2000}) Multinom(ni, (p_{i1}, p_{i2}, ..., p_{i1999}, p_{i2000})).$$

  Thus, we arranged our data as 233286 rows of users and columns of user features, as well as 2000 columns of play counts for different artists. Then we would fit the model in this way, and predict the play counts for new pairs based on its user's distribution (we would use 2000*user-median as the total number). However, due to the high model complexity and constraints of computing resources, the model fails to be fitted.

- Poisson log-linear model:

  By playing with the $use - median$ and analyzing the play counts of every user, we found that except for a few outliers, one user tends to have similar play counts for every artist he/she has listened to. This enlightened us to fit a Poisson log-linear model. Every user with its features can be treated as an observation, for example, let $x_{i1}, ..., x_{iJ}$ be the predictor variables of the i-th observation, we would assume a response variable Y_i(here Y_i

Model	MAE(10,000 User Sample)	MAE(Entire Sample)	Kaggle Score
LASSO REGRESSION	232.90952	—	—
RIDGE REGRESSION	232.93265	—	—
DECISION TREE	246.13667	—	—
RANDOM FOREST	241.22268	—	—
ENSEMBLE METHOD	—	—	—
K-MEANS USER CLUSTERING	—	200.29547	207.35844
MULTINOMIAL LOGIT MODELS	—	—	—
POISSON LOG-LINEAR MODEL	133.13023	—	206.62659
REVISED BASELINE 1	—	138.31546	137.65831
REVISED BASELINE 2	—	138.10270	137.40276

Table 1: Models with corresponding MAE score under subset and the entire train set.

is the play counts for this user) where $YiPo(\mu_i)$. The probability mass function is given by $p_i(y_i) = \frac{\exp^{-\mu_i}\mu_i^{y_i}}{y_i!}$ where $y_i \geq 0$. The predictors and the response variable is related with a link function as: $\log \mu_i = \mathbf{x}'\boldsymbol{\beta}$

Thus, we arranged our data as 233286 rows of users and columns of user features, as well as a column of each user's median play counts. We successfully fitted the Poisson log-linear model and made predictions on the test file. The final score on Kaggle is 206.62659, which failed to beat the baseline. The result is understandable because most play counts are fairly large(more than 100), so the response variables tend to be normally distributed rather than poisson distribution. Thus the Poisson log-linear model is not very appropriate here.

- Revised baseline 1:$alpha \times (user\_median)$

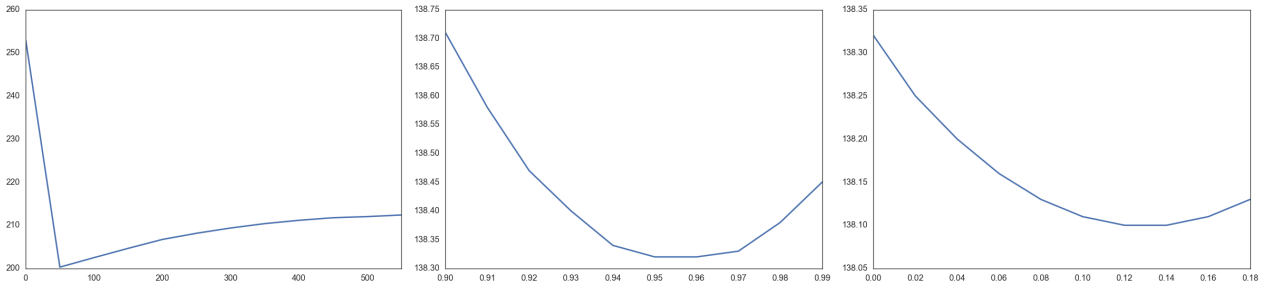
  We slightly changed the baseline model to estimate the number of plays when a user encounters a new artist. The baseline model performances well by directly using user median, but the true number might be float. So we added one more coefficient alpha to adjust the prediction and got the best alpha as 0.96 after cross validation. This approach results in a MAE of 138.31546 on the training set and 137.65831 on Kaggle.

- Revised baseline 2: $alpha \times (user\_median) + beta \times (artist\_median - global\_median)$

  We adjusted the revised baseline 1 model by taking the artist popularity into account. Obviously, a normal user will tend to listen to a popular artist's song more frequently. Therefore, We added a correction term to the previous equation, which is the product of the indicator of a artist's popularity($artist\_median - global\_median$) and a coefficient calculated via cross validation. This approach results in a MAE of 138.10727 on the training set with alpha = 0.96 and beta = 0.14 and 137.40276 on Kaggle.

2 Results

We have created and submitted a set of predictions on Kaggle. After trying both feature engineering and model selection, our final method gives reasonably good performance. The tuning plot of K-means, revised baseline 1 and revised baseline 2 is showing below. The summary of our results can be seen in Table 1. The revised baseline 2 produced the best result of 137.40276.



. K-means best MAE at K = 50 Revised baseline 1 best MAE at $\alpha = 0.96$ Revised baseline 2 best MAE at $\beta = 0.14$

3 Discussion

After a quick glance at the given datasets, we noticed that we had the information about users (”sex”, ”age”, ”country”) but almost no information about artists. This drove us to extract some useful artists’ attributes via musicbrainz API, and we successfully got features like ”country”, ”type”, etc. Naturally, we did some feature engineering (e.g. one-hot encoding). Then the original training data were converted to a giant feature matrix for user-artist pairs. With all the play counts, it appeared that this problem could be treated as a simple regression problem. So we first applied some regression models. However, it turned out that all the regression models we’ve tried are super computational-expensive, and performed very poorly on the subset. Therefore, we abandoned our original thoughts and started towards another direction.

On thinking that clustering helps to reduce the dimension and some users might present similar behavior patterns, we decided to try K-means clustering to group users. Experiments showed that K-means would run much faster than the above regression models. After several tries, we found that this method performed better when k is smaller. We also noticed that for a given user, the play counts he/she listens to different artists are similiar. This lead us to think of ignoring artists’ features since users appear to be more significant.

We then split into two working directions. The first direction, is to focus on the users’ features. We first thought that different users may have different multinomial distributions over artists’ play counts, so we tried to fit a multinomial logit model. However, due to the model complexity and large data amount, the model failed to be fitted. Then because each user has similiar play counts on different artists, we thought the play counts for each user may have a poisson distribution. So we fitted a poisson log-linear model. The final result failed to beat the baseline; the reason may be that large counts tend to diverge from poisson towards normal distribution.

Another working direction is to use user-median and adjust the baseline model. The fact that each user has similar play counts on different artists made us believe that user median is already a quick and good guess; true play counts might not be too far away. So we used coefficient α to adjust the prediction. We used cross validation to get the best estimated α to be 0.96. Next we also took the artist median into account (artist median$-$global median) with a coefficient β attained by cross validation to further improve the baseline model. Our models successfully beat the baseline and the second revised version got the best performance.

In short, we explored some regression models and clustering methods, but abandoned them due to poor performance. We then focused mainly on user behavior patterns, tried Multinomial logit and Poisson log-linear model, and also came up with revised baseline models. Our second revised user median model works the best and finally got MAE score of 137.40276 on Kaggle.