SYMA REPORT

Alexandre Lemonnier - Sarah Gutierez
Victor Simonin - Alexandre Poignant

Promotion 2023

Contents

1 Introduction

The objective of the project is to improve a recommendation system with application to the dressipi challenge. The goal was to obtain the best possible score on the online challenge, and to try different way to have a good score. The report will go through our analysis of the data, then on the different process and alteration that we have done to produce our final dataset, and finally on our different machine learning technique to generate our submissions.

2 Data Analysis

For this challenge we had access to multiple files that gave us many information about the behaviour of the customers. There were 5 files in total:

- 'candidate_items.csv': contains all the items available
- 'item_features.csv': contains all the features of each item
- 'train_purchases.csv': contains all the purchases that occurred at the end of a session.
- 'train_sessions.csv': contains all the items viewed in a session for each session_id
- 'test_leaderboard_sessions': contains the input sessions for the leader-board

First we took a look at the features' category of the items to see what was the most important ones. It's an interesting approach because we want to focus on the features that items have in common but if we remove some of them we might also loose some very specific case of recommendation.

Then we checked that none of the features were useless, and we saw that the feature number '27' had always the same value and was used only one time. We decided that it wasn't necessary to keep it and removed it.

Moreover, we checked how sessions are represented in the *train_sessions* dataset, and we observed that most of the sessions visit only one item. We saw too that each session bought only one item in the *train_purchases* dataset.

3 Data pre-processing

After this analyse, we had to pre-processed the data in order to used them in our recommendation systems. The *items* used 73 category of features. Even if 73 category is not a lot, it is still a big number and we had to reduce the dimension of our data to apply ML algorithm later. We used a truncated SVD to reduce our items' sparse matrix to 12 components. This matrix allowed us to find easily and faster the embedding items of each items. We had just to compare the value of their components in the matrix.

Secondly, regarding the sessions' data, we had a lot of data, so we decided to use only 10% of the *train_sessions* dataset. After that, we ranked the sessions by their number of items. To reduce more the data, we fixed a threshold to 90% of sessions by doing a cumulative sum in the rank's descending order. We found that we only needed to keep the data of sessions that visited at most 3 items to got 90% of the sample's sessions. By joining the sample's data with the matrix to represent the embedding items, we had now a dataset with the features' information of the items each session visited.

Obviously, we had to prepare the data representing the purchases of the sessions. To do so, we labelled 4 copy of the *train_purchases* dataset as not purchase and suffled their *item_id*, and merge them with a copy of the original dataset, labelized as bought. Finally, we concat to this dataset the SVD matrix to get the information of the features of the purchased items.

Finally, we merge the whole dataset together to fit a model. For each session we concatenate:

- The history of the user
- The purchase embedding, with the label

4 Recommend items

In order to recommend items, we decided to go through all the candidate's items and predict the probability of buying the item regarding to the session' items history. To do so, we got the item's features values, and we create an input for our model.

Finally, we have just to order them by their probability and take the number of items to recommend.

5 Machine Learning

For the machine learning part, we used two different models to generate our submission. The first one is a logistic regression and the second one is a simple RNN (Recursive Neural Network).

When evaluating the models on a test dataset, the logistic regression gave us a decent accuracy score of 79,99% , whereas the RNN gave us a quite better score of 80,91%.

6 Improvements

To improve our results, we thought that removing some features could be a solution. So we decided to use only the 15 most used features' category and retry our experimentation. We got as result an accuracy of 80,00% with the logistic regression and 80,92% with the RNN. We concluded that it does not really improve the performance of our predictions, because it could be based on the sample used as data, that is generated randomly.

7 Leaderboard

To compute the final submissions used for the RecSys Challenge 2022, we had to perform our predictions for all the sessions in the dataset and pick the hundred items with the highest probability to be bought and save the results in a .csv file. The complexity of the algorithm was too high to be compute with one computer, resulting in several dozen hours of computation. To get some results, we decided to focus on the 150 most popular candidate items in the *train_sessions* dataset, meaning the most visited items. After around 6 hours of processing, we got a score of 17,80% as the team 'NOOT-NOOT!!!' on the RecSys Challenge leaderboard.