Technische Universiteit Delft
Faculty of Electrical Engineering, Mathematics and Computer Science

Netflix Challenge: Movie Rating Prediction

CSE-2525 Data Mining
Thomas Abeel, Gosia Migut

Prepared by
Yanqing Wu
Student ID 5142571
Kaggle ID yanqingwutudelft
Exchanged Computer Engineering
1 January 2020

Table of Contents

List of Figures

List of Tables

1 Introduction

The report, entitled ‘‘Netflix Challenge: Movie Rating Prediction’’, is prepared as my Challenge report for the course CSE2525-Data Mining at the Technische Universiteit Delft. The purpose of this report is to develop a recommendation system for predicting movie ratings. The goal of the recommendation system is to achieve Root Mean Square Error (RMSE) as small as possible on an unseen dataset.

1.1 Netflix Datasets

Table 1-1. The Basic Information of Provided Data Sets

Dataset	Features	Mean	Std	Min & Max
users	gender	0.72	0.45	0.00 1.00
	age	30.64	12.90	1.00 56.00
	profession	8.15	6.33	0.00 20.00
movies	year	1985.81	16.91	1919.00 2000.00
	title (string)	-	-	-
ratings	rating	3.58	1.12	1.00 5.00

In users - ‘gender’, ‘0’ and ‘1’ indicates female users and male users, respectively;
In movies - ‘year’, only non-zero entries are considered.

Three datasets are provided for training, as described in Table 1-1. There is a total of 910,190 ratings, which were given by 6,040 users and 3,706 movies. The rest ‘predictions.csv’ file is used for final testing, which contains only ‘userID‘ and ‘movieID’ for each entry. There is a total of 90,019 entries in ‘predictions.csv’ to be predicted and the result is put on Kaggle to determine RMSE score.

2 Methodology

Data interpretation was first conducted at the beginning of the Challenge, as demonstrated in Table 1-1. After examining the data and reviewing the course materials, the first version of the recommendation algorithm was decided as Collaborative Filtering (CF). More specifically, Item-Item CF was selected as the first attempt to the Challenge. Item-Item CF is more reliable than User-User CF in practice [2], because items are much less dynamic than users.

2.1 Item-Item Collaborative Filtering

Collaborative filtering is an approach of ‘‘making automatic predictions (filtering) about the interest of a user by collecting preferences information from many users (collaborating)’’ [1]. There are usually two approaches for collaborative filtering (CF), namely Item-Item CF and User-User CF. User-User CF is generally more difficult to scale than Item-Item CF due to the dynamic nature of users, whereas items usually remains constant. Hence, Item-item CF was selected.

2.1.1 Core Equation

$$\begin{aligned} r_{xi} &= b_{xi} + \frac{\sum\limits_{j \in N(i;x)} S_{ij} \cdot (r_{xj} - b_{xj})}{\sum\limits_{j \in N(i;x)} S_{ij}} \\ b_{xi} &= \mu + b_x + b_i \end{aligned} \tag{1}$$

Equation (1) shows the core idea of the algorithm [3]. Term r_{xi} represents the rating of user x on movie i. Term s_{ij} is the similarity of movie i and movie j, which is measured by centered cosine similarity. Term b_{xi} is the baseline estimator for r_{xi}, where μ is the average ratings of all movies, b_x is the rating deviation of user x and b_i is the rating deviation of movie i.

2.1.2 Nearest Neighbors

For each prediction, the top similar movies to movie i need to be determined, so that a more reasonable prediction can be made. The top similar movies to movie i is known as the *nearest neighbors* of item i. A parameter N is used to denote the amount of nearest neighbors in the following context. Additionally, Jaccard similarity is abandoned in determining similarity since it only considers common movies rather than ratings. Moreover, Cosine similarity has a more informed interpretation than Jaccard similarity; however, it fails to generalize among users since different users have different standards. Pearson correlation (a.k.a centered cosine similarity) preserves the advantage of cosine similarity and removes the bias views of users via subtracting row mean for each movie.

An upper triangular matrix of similarity score is built to fulfill all aforementioned points.

Table 3-1. N and Cap. vs. RMSE

N	Cap.	RMSE
5	off	0.88071
7	off	0.86398
10	off	0.85458
10	on	0.85397
12	on	0.85144
15	on	0.84996

3 Results

Table 3-1 shows the result of the recommendation system (described in 2.1) on Kaggle test set. As mentioned in 2.1.2, *N* is the amount of nearest neighbors.

After noticing some negative ratings and nearly 6.0 ratings, option *Cap.* is introduced. The *Cap.* option sets ratings that below 1.0 to 1.0 and ratings that above 5.0 to 5.0.

The maximum static *N* is 15, because the minimum amount of user *i* rated movies is 15.

Table 3-2. Rounding vs. RMSE (*Cap.* = on)

N	ε	RMSE
10	0.1	0.85458
10	0.05	0.85409
10	0.02	0.85398
10	0.01	0.85398
10	0	0.85397

Table 3-2 shows an attempt to use rounding to improve RMSE. Hyper-parameter ε represents the range of rounding. For instance, $\varepsilon = 0.05$ rounds any ratings in the range of $[3.95, 4.05]$ to a 4.0. Since the final rating is integer, the author had a mis-belief that rounding those almost certain ratings to their nearest integers would enhance the final result. Nevertheless, the result did not improve.

Table 3-3. Dynamic N vs. RMSE (*Cap.* = on)

Dynamic N	RMSE
L_x	0.87736
min(round(0.1L_x), 15)	0.85761
min(round(0.2L_x), 20)	0.85280
min(round(0.3L_x), 30)	0.85307
max(round(0.1L_x), 5)	0.85754
max(round(0.2L_x), 5)	0.86148

Table 3-3 shows an attempt to use dynamic N to improve RMSE. Parameter L_x is the amount of user *x* watched movies.

4 Discussion

Cold-start problem and dealing with missing entries will be discussed in XXX section.

5 Conclusions and Future Work

From the analysis in the report body, it was concluded that...

References

[1] *Collaborative filtering*. 2019. URL: `https://en.wikipedia.org/wiki/Collaborative_filtering`.

[2] J. Leskovec, A. Rajaraman, and J. D. Ullman. *Mining of Massive Datasets*. New York: Cambridge University Press, 2009.

[3] *Mining of Massive Datasets*. URL: `http://mmds.org/`.