Project No: SCSE20-0248

A Performance Analysis on Non-Personalized Recommender Systems (Interim Report)

Submitted by: Li Guanlong

Matriculation Number: U1722033H

Supervisor: Assoc Prof Sun Aixin

School of Computer Science and Engineering

2021

Chapter 1 Introduction

1.1 Motivations

Recommender system is an effective tool to deal with information overload. It is adopted in different businesses such as video streaming, e-commerce and social media to provide their user with a better experience.

1.2 Objectives and Scope

This project utilized dataset provided by MovieLens, which is a website where users can rate movies in a scale of one to five stars. In this project, all algorithms are implemented and evaluated against MovieLens dataset.

In this project, recommender systems are considered to have two categories, which are (i) personalized and (ii) non-personalized. This project aims to provide an analysis on non-personalized recommender systems with the MovieLens dataset. The analysis includes (i) an in-depth overview of existing commonly used algorithms which are namely MostPop, RecentPop, DecayPop, and (ii) a new algorithm which is based on the timeline of each individual movie. With the analysis on the performance of different algorithms, this project also aims to show the effectiveness of this new algorithm. All the algorithms discussed in this project are popularity based. Popularity of a movie is defined as the number of user interactions received by the movie.

For evaluation of the algorithms, the number of stars that user gives in a rating is not considered as a metric. Instead, the evaluation only focuses on whether the user performs the action of rating, i.e., a user-movie interaction. For example, if the system makes a prediction on movies' popularity on day t_0 based on the data up to day $t_0 - 1$. Then, it recommends a movie m to user u based on the popularity predicted. In other word, the recommender predicts that user u will rate movie m on day t_0. In this case, this recommendation is considered to be a correct one for user u as long as user u does rate movie m on day t_0, regardless of the number of starts that the user gives. R-precision is chosen to be the measurement for evaluating the performance since all the algorithms discussed in this project are essentially top-N recommender.

1.3 Algorithm Experimented

1.3.1 **MostPop**

MostPop makes recommendation on t_0 based on the total number of interactions a movie received from users in the training set up to t_0 - 1.

1.3.2 **RecentPop**

RecentPop focuses on the recent interactions by introducing a Δt such that the recommendation on t_0 is based on the total number of interactions a movie received from user during time period $[t_0 - \Delta t, t_0 - 1]$ in the training set.

1.3.3 **DecayPop**

DecayPop introduces a decay parameter on top of RecentPop. In DecayPop, the calculation is a weighted sum of interactions, with more recent interaction carries a higher weight. For instance, when making recommendation on t_0 given that Δt is n days and in these n days, movie m is interacted for i_1, i_2, i_3 … i_n times respectively, where i_1 is corresponding to $t_0 - 1$ (the most recent day) and i_n is corresponding to $t_0 - \Delta t$ (the least recent day). Then the recommendation is based on the value of $\sum_{t=1}^{n} e^{-t} \cdot i_t$ calculated for each movie.

1.3.4 **Timeline-Based Algorithm**

This algorithm is a new algorithm defined in this project. The timeline of a movie refers to the number of interactions it received on each day of the past time period. For example, if a movie m is rated for [7, 6, 5, 4, 3, 2, 1] times during the past 7 days, where the right side indicates most recent, and left side indicates least recent. Then, this array is called the timeline of m during past 7 days. Suppose the most recent day is $t_0 - 1$, and the recommendation is for t_0. The difference between this timeline-based algorithm and the other three algorithms is that the other algorithms are summation based, such that the predicated popularity of m on t_0 would be a sum or weighted sum of elements in the array (if $\Delta t = 7$), or a sum of all the interactions received by m up to $t_0 - 1$. Meanwhile, the timeline-based approach considers this array as something continuous and focuses on the trend of the timeline. A simple example of such algorithm would be to predict the popularity of m on t_0 to be 0, since the numbers of interaction have a trend to decrease by 1 per day. This algorithm is still being experimented and the details of this algorithm is still subject to change.

Chapter 2 Data and Result

2.1 Experiment Setup

MovieLens dataset which is used in this project contains the user-movie interaction data between January 09, 1995 and November 21, 2019. Since recent data is a better representation of actual user behaviour in recent time, only the data in the most recent 10 years were used after dropping the last incomplete month. The data between November 01, 2009 and October 31, 2014 were used as training set, which was to train MostPop, and was also effective for the first Δt days for RecentPop and DecayPop. The data between November 01, 2014 and October 31, 2019 were used as evaluation set. The evaluation was done on a daily basis. That is to say, MostPop, RecentPop, DecayPop, and timeline-based algorithms were run on every day to produce a top-N recommendation for that day. Since the algorithms discussed in this project are non-personalized algorithm, the same top-N recommendations were provided to all the users. Then based on this recommendation and the actual movies rated by different users, a daily R-precision was calculated for each day. Then, monthly average and yearly average were calculated in each year and month interval.

2.2 Algorithm Exploration Results

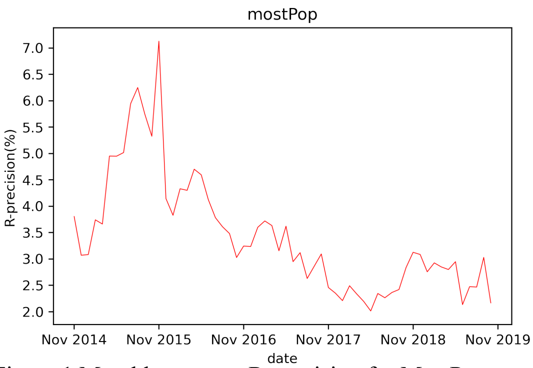

Figure 1 Monthly average R-precision for MostPop

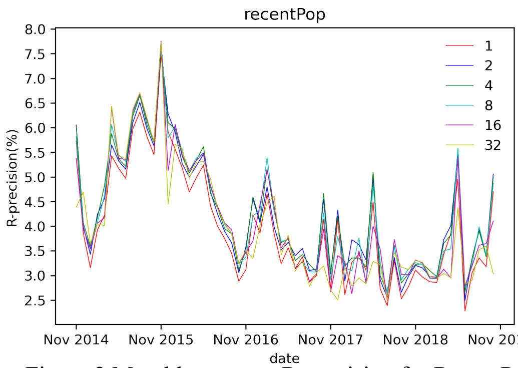

Figure 2 Monthly average R-precision for RecentPop

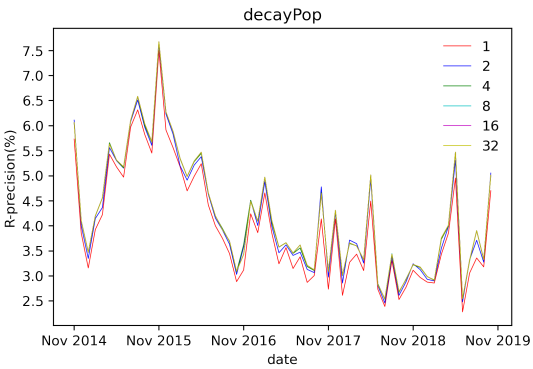

Figure 3 Monthly average R-precision for DecayPop.

mostPop R-precision(%)

	Nov 2015	Nov 2016	Nov 2017	Nov 2018	Nov 2019
1-32	4.630	4.244	3.234	2.354	2.725

Table 1 Yearly average R-precision for MostPop

recentPop R-precision(%)

	Nov 2015	Nov 2016	Nov 2017	Nov 2018	Nov 2019
1	5.005	4.791	3.584	3.131	3.392
2	5.222	5.016	3.823	3.368	3.655
4	5.320	5.089	3.894	3.388	3.665
8	5.310	5.107	3.814	3.334	3.610
16	5.265	5.059	3.722	3.185	3.447
32	5.256	4.946	3.546	2.975	3.232

Table 2 Yearly average R-precision for RecentPop

decayPop R-precision(%)

	Nov 2015	Nov 2016	Nov 2017	Nov 2018	Nov 2019
1	5.005	4.791	3.584	3.131	3.392
2	5.180	4.973	3.819	3.317	3.581
4	5.239	5.029	3.865	3.367	3.639
8	5.244	5.035	3.858	3.372	3.644
16	5.244	5.035	3.858	3.371	3.645
32	5.244	5.035	3.858	3.371	3.645

Table 3 Yearly average R-precision for DecayPop

Figure 1, 2, 3 shown above indicates the monthly average of R-precision for MostPop, RecentPop and DecayPop during evaluation. Similarly, Table 1, 2, 3 indicates the yearly average. For RecentPop and DecayPop, different values of Δt between 1 day and 32 days, were experimented. For the evaluation set which consists of the last 5 years of data, the separator between any two years is October 31 and November 01. For simplicity, in the rest of this report, year 2015 represents the time between Nov 2014 and Nov 2015, similarly for year 2016, 2017, 2018, 2019

From above figures and tables, it can be concluded that (i) The optimal Δt was 4 days for RecentPop, and 8 days for DecayPop, (ii) when using optimal Δt, RecentPop outperformed DecayPop and MostPop and (iii) R-precision calculated had a tendency to reduce as time moved towards a more recent time. In the rest of this report, Δt = 4 days is used for RecentPop, Δt = 8 days is used for DecayPop unless otherwise stated.

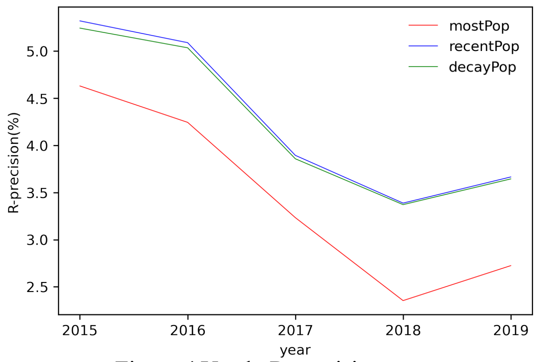

Figure 4 Yearly R-precision pattern

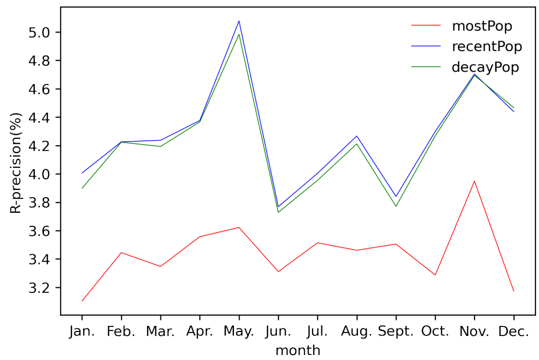

Figure 5 Monthly R-precision pattern

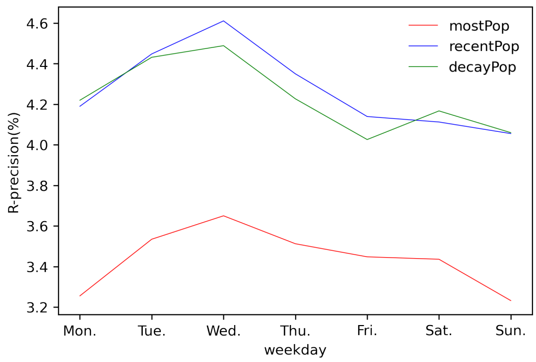

Figure 6 Weekly R-precision pattern

Figure 4, 5, 6 shown above indicates the yearly, monthly and weekly pattern of R-precision calculated based on the evaluation set. Yearly, monthly and weekly pattern was calculated by taking average value of all the daily R-precision lies in each specific year, month and weekday, respectively. From above figures, it can be concluded that (i) apart from the small increase of yearly R-precision on 2019, the R-precision had a tendency to drop as time moved towards a more recent time, which confirms with the observation from Figure 1, 2, 3, (ii) MostPop, RecentPop and DecayPop generally had similar pattern, (iii) monthly R-precision pattern had two local maxima on May and November, and two minima on January and June, and (iv) weekly R-precision pattern reached maxima on Wednesday and reached minima on Monday and Sunday.

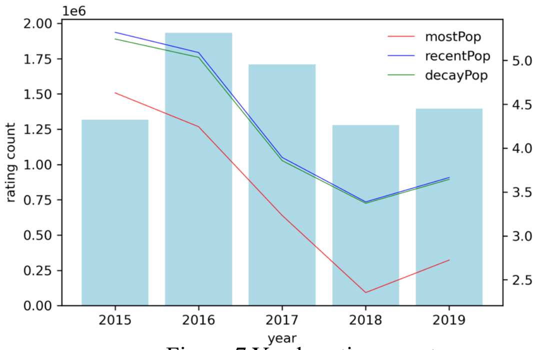

Figure 7 Yearly rating count

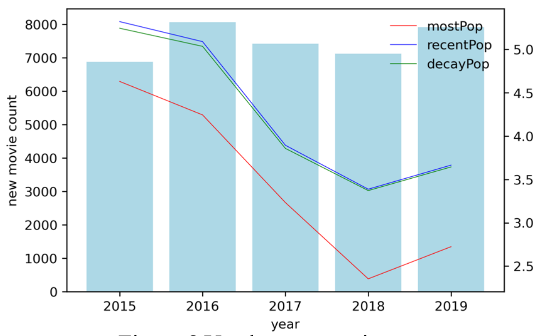

Figure 8 Yearly new movie count

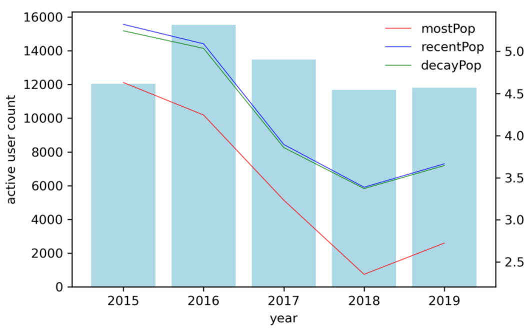

Figure 9 Yearly active user count

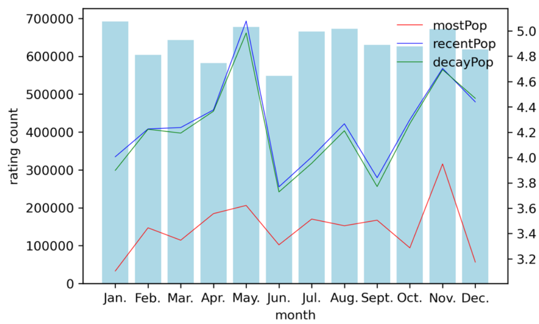

Figure 10 Monthly rating count

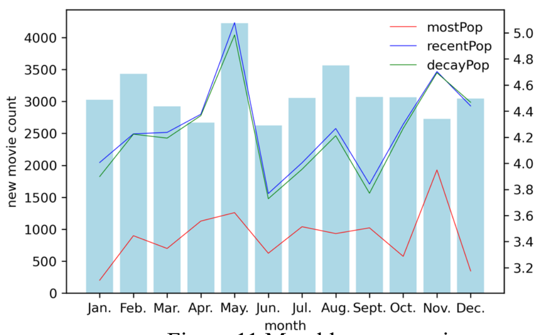

Figure 11 Monthly new movie count

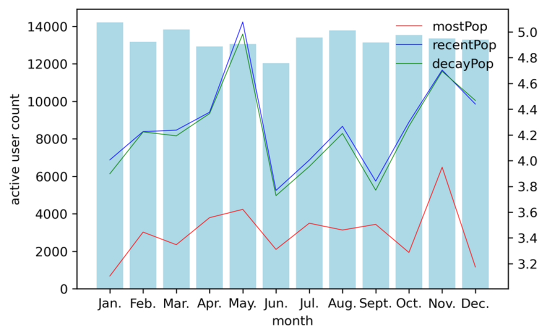

Figure 12 Monthly active user count

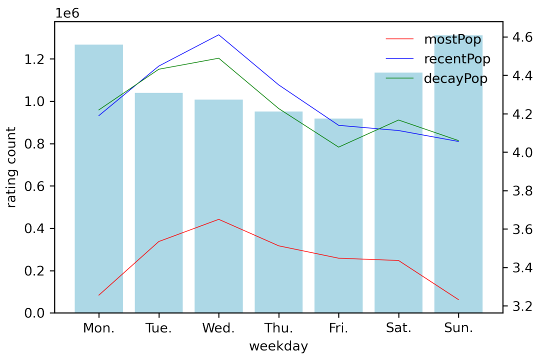

Figure 13 Weekly rating count

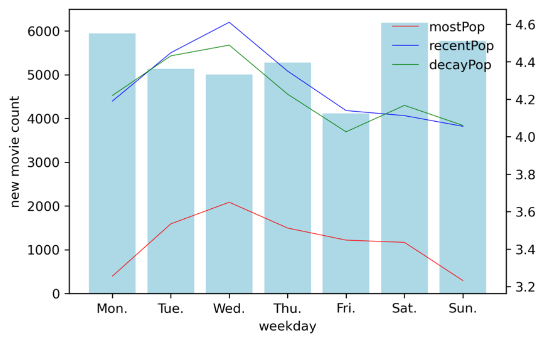

Figure 14 Weekly new movie count

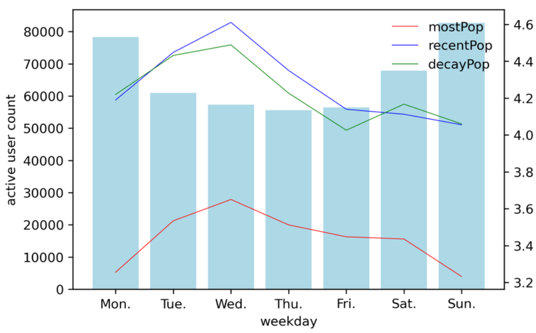

Figure 15 Weekly active user count

In order to find the reasons behind the R-precision patterns, above figures were plotted. Figure 7, 10, 13 shows the total number of ratings, i.e., user-item interactions, that falls in each year, month and weekday. Figure 8, 11, 14 shows the total number of new movies that are induced in each year, month and weekday. Figure 9, 12, 15 shows the total number of active users that are presented in each year, month, and weekday.

Active user was defined as user who provided at least 1 rating during the specific year, month, or weekday. For a clearer comparison, yearly, monthly, weekly R-precision patterns were also plotted in yearly, monthly, weekly associated figures.

From the figures, it can be concluded that (i) yearly and monthly patterns did not show an obvious correlation with the rating count, new movie count or active user count of the year and month, (ii) weekday pattern showed a negative correlation with the rating count and active user count. The two weekly R-precision minima, Sunday and Monday were also the two weekdays that had largest number of rating and active user.

2.3 Timeline-Based Algorithm

2.3.1 Pre-Selection

Since timeline-based algorithm is computationally expensive because it performs one set of operations per movie. There is a need for pre-selection in order to find a subset of the available movies which can reduce the computation cost while having a relatively small impact on the performance. The pre-selection will be done by setting a threshold based on the number of ratings a movie received during each of the past 7 days. This is because more popular movie carries a higher value for recommendation.

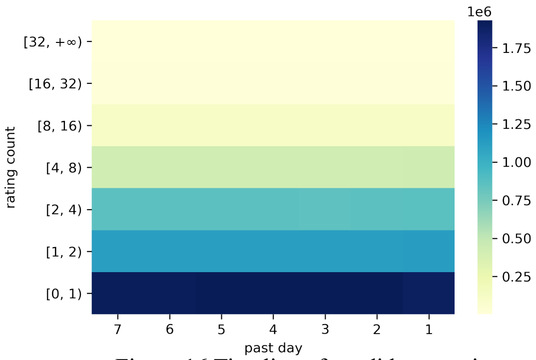

Figure 16 Timeline of candidate movies

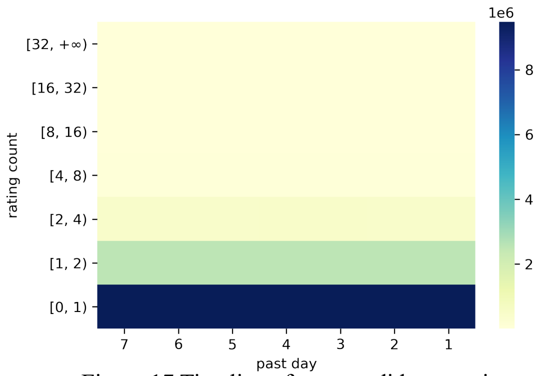

Figure 17 Timeline of non-candidate movies

In above Figure 16 and 17, a sliding window method was applied to visualize the past 7-day timeline of candidate and non-candidate movies in two heatmaps. From the 8^{th} day until the last day of the 10-year data, suppose each individual day was called current day in its own sliding window. In this case, the sliding window contained 8 days, including the current day and the previous 7 days. The sliding window moved 1 day per step, then, candidate movies were defined as the movies which had been rated at least once during the previous 7 days and were also rated on the current day. On the other hand, non-candidate movies were defined as the movies which had been rated at least once during the previous 7 days but were not rated on the current day. Figure 16 and 17 were plotted by identifying candidate movies and non-candidate movies in each sliding window and visualizing their timeline in the previous 7 days. Figure 16 visualizes the timeline of all the candidate movies in all the sliding windows, Figure 17 visualizes the timeline of all the non-candidate movies in all the sliding windows. For both figures, x-axis indicates the timeline, 7 indicates 7 days before current day, and 1 indicates 1 day before the current day; y-axis indicates the range of rating count a movie received per day per sliding window; the colour in heatmap indicates the occurrence number of a timeline having a rating count which falls in the range indicated by y-axis on the day indicated by x-axis in a sliding window.

From the two figures, it can be concluded that, for candidate movies, the percentage of movies that had been rated for at least (higher or equal to) 4 times in any of the previous 7 days was noticeable. Meanwhile, for non-candidate movies, majority of them had been rated for less than 4 time in any of the previous 7 days. Therefore, a potential criterion for making the pre-selection is that the movie must be rated for at least 4 times in any of the 7 previous days.

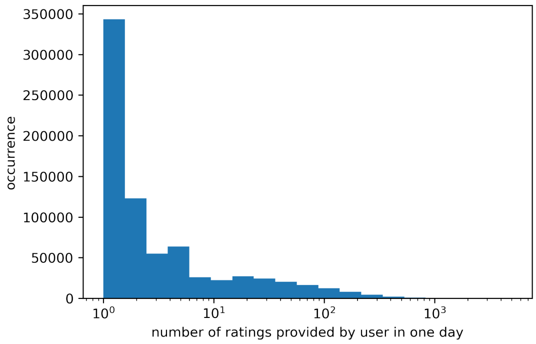

Figure 18 Rating count per user per day

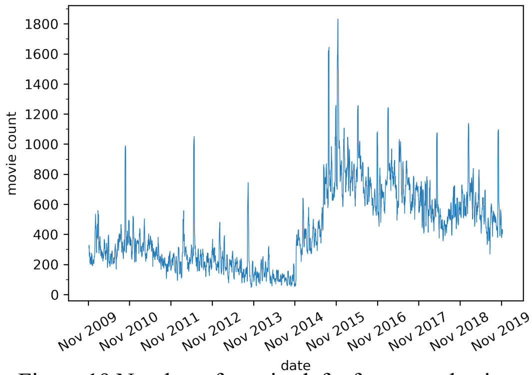

Figure 19 Number of movies left after pre-selection

Suppose user daily activity refers to the number of ratings a user provided in a day. Then, Figure 18 shows the distribution of all the user daily activity on every day throughout the 10 years. X-axis is in log scale and indicates the number of ratings a user provided in a day; y-axis indicates the occurrence number of such user daily activity throughout the 10 years.

Figure 19 shows the number of possible candidate movies left on each day after the pre-selection criterion "the movie must be rated for at least 4 times in any of the 7 previous days" was applied. Possible candidate movie was defined as the movie which had been rated for at least once during the previous 7-day $[t_0 - 7, t_0 - 1]$.

From Figure 18 and 19, it can be concluded that throughout the 10 years, vast majority of users provided less than 10 ratings daily. Meanwhile, after applying the criterion stated above, vast majority of days had a subset of possible candidate movies which fit this criterion while containing at least 100 to 200 movies. Therefore, the threshold of 4 is feasible since after applying this threshold to obtain a subset of movies, by applying the timeline-based algorithm, there would still be sufficient movies left to recommend to vast majority of users.

2.3.2 Timeline-Based Algorithm Implementation

A simple linear regression was applied to show a prototype of timeline-based algorithm. More complex algorithm will be experimented and implemented later. This algorithm was run in a way such that for each day t_0 in the evaluation set, possible candidate movies were obtained. Then, a pre-selection was done to only keep the movies which had been rated for at least 4 times in any of the 7 previous days among possible candidate movies. Such subset of movies was called pre-selected possible candidate movies. After that, for each of the pre-selected possible candidate movies, the previous 7-day $[t_0 - 7, t_0 - 1]$ timeline of the movie was used to train a linear model, which was then used to predict the popularity of that specific movie on t_0.

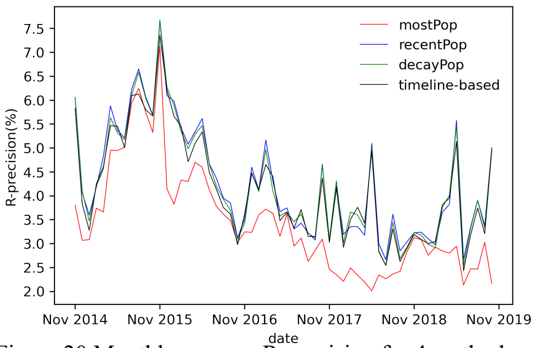

Figure 20 Monthly average R-precision for 4 methods

	Nov 2015	Nov 2016	Nov 2017	Nov 2018	Nov 2019
mostPop	4.630	4.244	3.234	2.354	2.725
recentPop	5.320	5.089	3.894	3.388	3.665
decayPop	5.244	5.035	3.858	3.372	3.644
timeline-based	5.118	4.893	3.839	3.332	3.568

Table 4 Yearly average R-precision for 4 methods

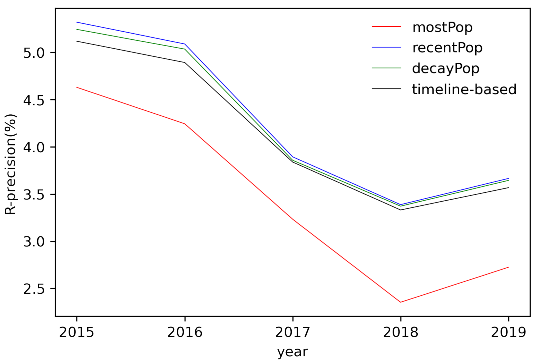

Figure 21 Yearly R-precision pattern for 4 methods

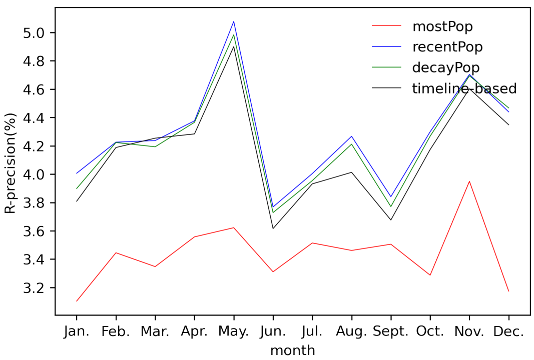

Figure 22 Monthly R-precision pattern for 4 methods

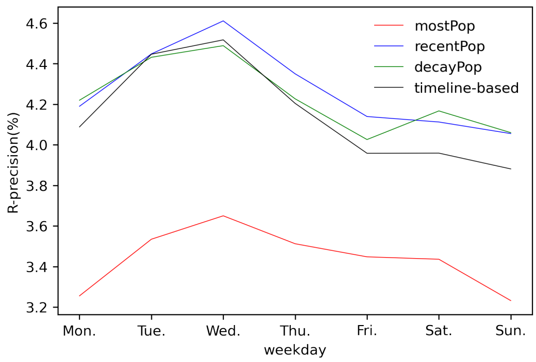

Figure 23 Weekly R-precision pattern for 4 methods

As shown in Figure 20, 21, 22, 23 and Table 4, the monthly and yearly average R-precision, and also the yearly, monthly and weekly R-precision patterns of the timeline-based method with linear regression were plotted together with those of MostPop, RecentPop and DecayPop for comparison. It can be concluded that (i) when using linear regression in the timeline-based method, the R-precision it achieved was lower than RecentPop and DecayPop but was significantly higher than MostPop. (ii) The yearly, monthly and weekly R-precision patterns of the timeline-based method is similar to the patterns of MostPop, RecentPop and DecayPop.

Chapter 3 Future Works

This interim report contains the current progress of the final year project up to 24 January 2021. For the rest of the current semester, some possible future works are listed in this chapter.

3.1 Addition on Timeline-Based Algorithm

The implementation of the timeline-based algorithm consists of two parts, (i) pre-selection of possible candidate movies and (ii) the algorithm which runs on the timeline data. In this interim report, the timeline-based algorithm implemented in section 2.3.2 is only a prototype to demonstrate the potential performance of the timeline-based algorithm. It was implemented using a threshold-based pre-selection and a simple linear regression algorithm. During the rest of the current semester, more ways of implementing the timeline-based algorithm will be experimented and analysed to improve its performance.

3.2 Literature Review

Literature review is not included in this interim report, but it will be an ongoing process and will be included in the final report.

3.3 Possibly Other Algorithms

Apart from MostPop, RecentPop, DecayPop and the timeline-based method, additional methods might be introduced to the project. One of the possible options is the matrix factorization approach. By introducing additional methods and comparing the methods with the 4 existing methods, there might be some new discoveries regarding the effectiveness of movie recommendation algorithms.