Movies Recommendation

Irene Azzini, Stefano Bracchi, Lúcia Moreira

26/5/2019

Introduction

The aim of this assignment is to compare the performance of different recommendation strategies. Training data was obtained from **RottenTomatoes**, a leading online aggregator of movies and TV shows reviews from different critics. We have used the **movies** dataset which consists of 1133 movies and the **reviews** dataset which consists of 51710 reviews made by 3496 different reviewers (users).
In this project, we have applied data mining tools such as **text mining and recommendation**, to learn different systems to recommend Top-N movies for a given user (critic/reviewer). Recommender systems apply statistical and knowledge discovery techniques to problems related to making recommendations based on previously recorded data [1]. Such recommendations can help critics to find movies they would probably like and can improve the reliability of the site **RottenTomatoes** creating a value-added relationship.

Task 1- Exploratory data analysis and pre-processing of the data

We have performed an **exploratory analysis** of our data in order to understand which type of data we have and how we can use it to create the required tables for the recommender systems.

DataSet

For this work we have used two datasets:

1. **movies.tsv**, that contains basic information about movies. The useful information is:

- *id*: id of the movies, useful to join the two datasets.

2. **reviews.tsv**, that contains information of the reviews made by critics for the movies they have seen. The useful informations are:

- *id*: id of the movies;
- *review*: text used in text mining for obtaining ratings where those are missing;
- *rating*: ratings given by the critics, with different scales of ratings;
- *fresh*: fresh if the critic likes the film, otherwise rotten (*Tomatometer score*);
- *critic*: name of the critics that made the reviews;
- *date*: date in which the reviews were written.

As explained above, text mining was used for improving the number of ratings available on the dataset since in 13517 cases critics wrote the review but did not provide an evaluation rate. This happens because in **RottenTomatoes** the critics are required to upload their reviews on the website and to mark their review as *fresh* or *rotten*, *fresh/rotten* ratings is mandatory while the numeric score is not. We have used *Sentiment Analysis* to infer a numerical rating from the text written and converted from a [-1, 1] scale to a [0, 10] scale related to negative, neutral or positive feelings in relation to the written text.

Analysis of the data

We did our work by doing both *binary* and *non-binary* analysis for what concern the ratings that the critics made.

As first things:

1. We have removed from **reviews** dataset the rows that correspond to an empty critic name, so those who did not have a critic's name;
2. From **movies** dataset, the rows that correspond to a film which did not have any reviews were also removed.

By doing these adjustments, we have obtained 51710 reviews and 1133 movies with 3496 critics to work with on the *binary* approach.

For what concerned the *non-binary* approach, using also the reviews obtained with *sentiment analysis*, we have obtained 51371 reviews and 1132 movies with 3489 critics to work with. For this approach we have created new datasets that contain only the reviews and the movies on which we can construct our models: **reviewsNONBIN** and **moviesNONBIN**, respectively.

Then for the *binary* analysis we have added to **reviews** dataset the column *dummiesfresh* that contains dummy variables where 1 corresponds to a *fresh* review and 0 to a *rotten* one.
For the *non-binary* analysis, first we have normalized the different scales of ratings using the conversion reported in *Table 1* and then we have added to **reviewsNONBIN** dataset the column *dummiesrev* that contains the ratings, in scale [0, 4], given by the critics.

Original value	**Converted value**
A(+,-), [5,6]/6, [5,4]/5, [10,9,8]/10	4
B(+,-), [3,4]/6, 3/5, [7,6]/10	3
C(+,-), 2/6, 2/5, [5,4]/10	2
D(+,-), 1/6, 1/5, [3,2]/10	1
E(+,-), 0/6, 0/5, [1,0]/10	0

Table 1 : Table for conversion from the different ratings to ratings in [0, 4] (all the double numbers are rounded up)

To start the study of our data we have calculated for each critic the number of reviews done, we have seen that of the 3496 critics, 2544 (almost 30% of them) made less than a total of 10 reviews while only 107 critics made more than 100 reviews. We report in *Figure 1* the plot of the number of reviews done by critics that made more than 100 reviews, from this figure it is possible to see that indeed most of the critics in this subset review around 100 - 200 films.

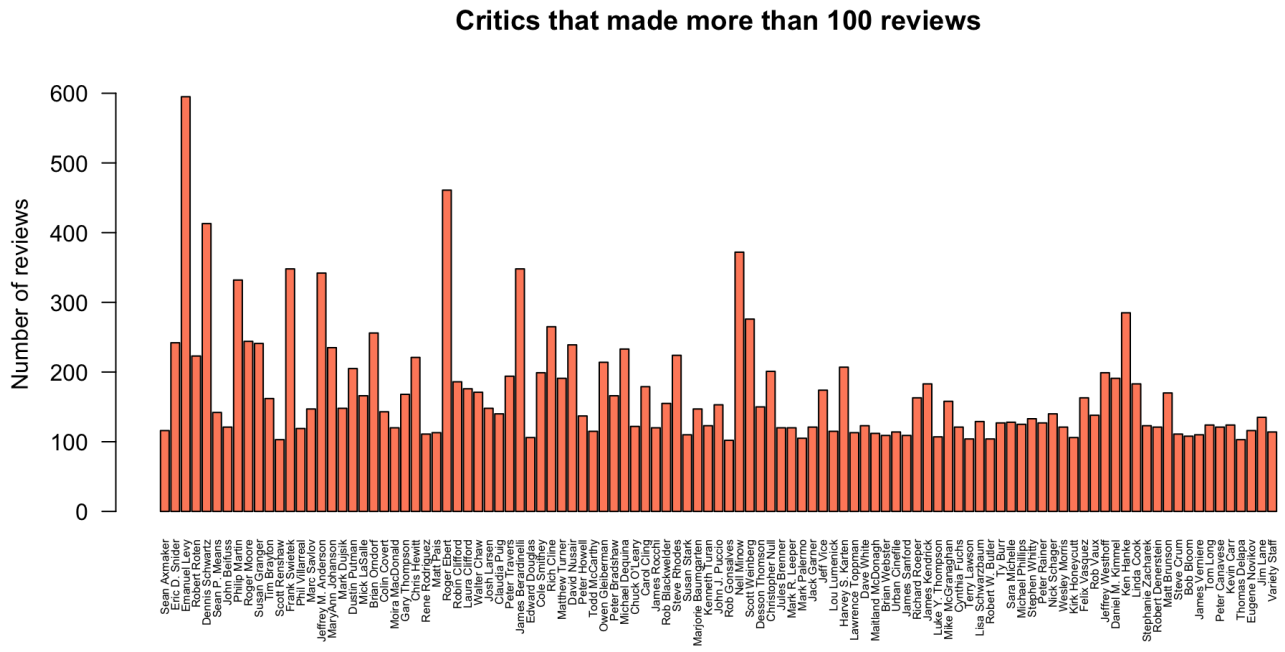

Figure 1 : Number of reviews done by critics that made more than 100 reviews

For those critics that have done more than 100 reviews, we have studied how they tend to evaluate the films, so if they tend to do more *fresh* or more *rotten* reviews. We have found out that most of them (80%) tend to evaluate the movies as *fresh* (more than 50% of the movies evaluated are *fresh*) (see *Figure 2*).

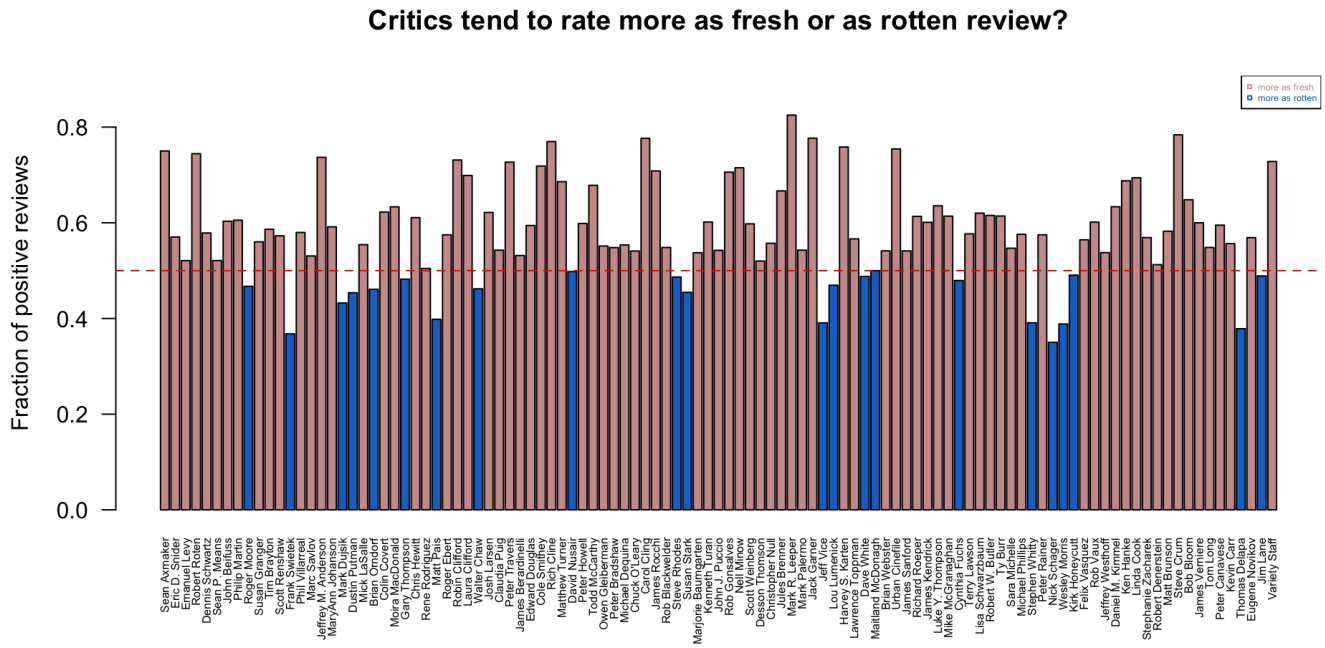

Figure 2 : Fraction of fresh reviews made by critics that made more than 100 reviews

From the plot in *Figure 3 (left)* it is possible to see that in general critics made *fresh* reviews and from the one in *Figure 3 (right)* we can see that for the non-binary ratings the most of the critics gave ratings in [2,3].

Figure 3 : Number of fresh and rotten reviews (right) and of the non-binary ratings (left)

We have considered also for each film how many times it was reviewed, by doing so we have found out that 69% of the movies have less than 50 evaluations while 12% have at least 120 evaluations. In *Figure 4* it is shown the number of evaluations made only for the films rated more than 120 times.

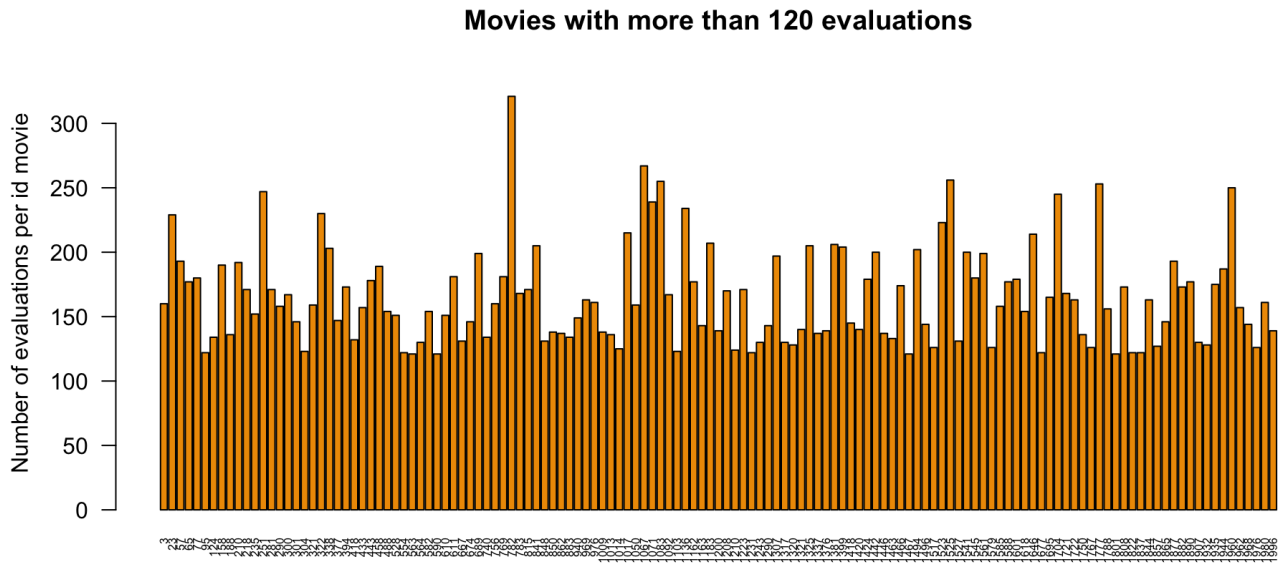

Figure 4 : Number of reviews for the film reviewed more than 120 times

Having these information about reviews by each critic and number of reviews for each movies we can infer that our dataset is sparse.

We have examined which is the most popular genre of the film seen by the critics and we have found out that most of the users watch *drama* or *comedy*, in *Figure 5* there is the plot of the number of films watched for each of the 21 genres.

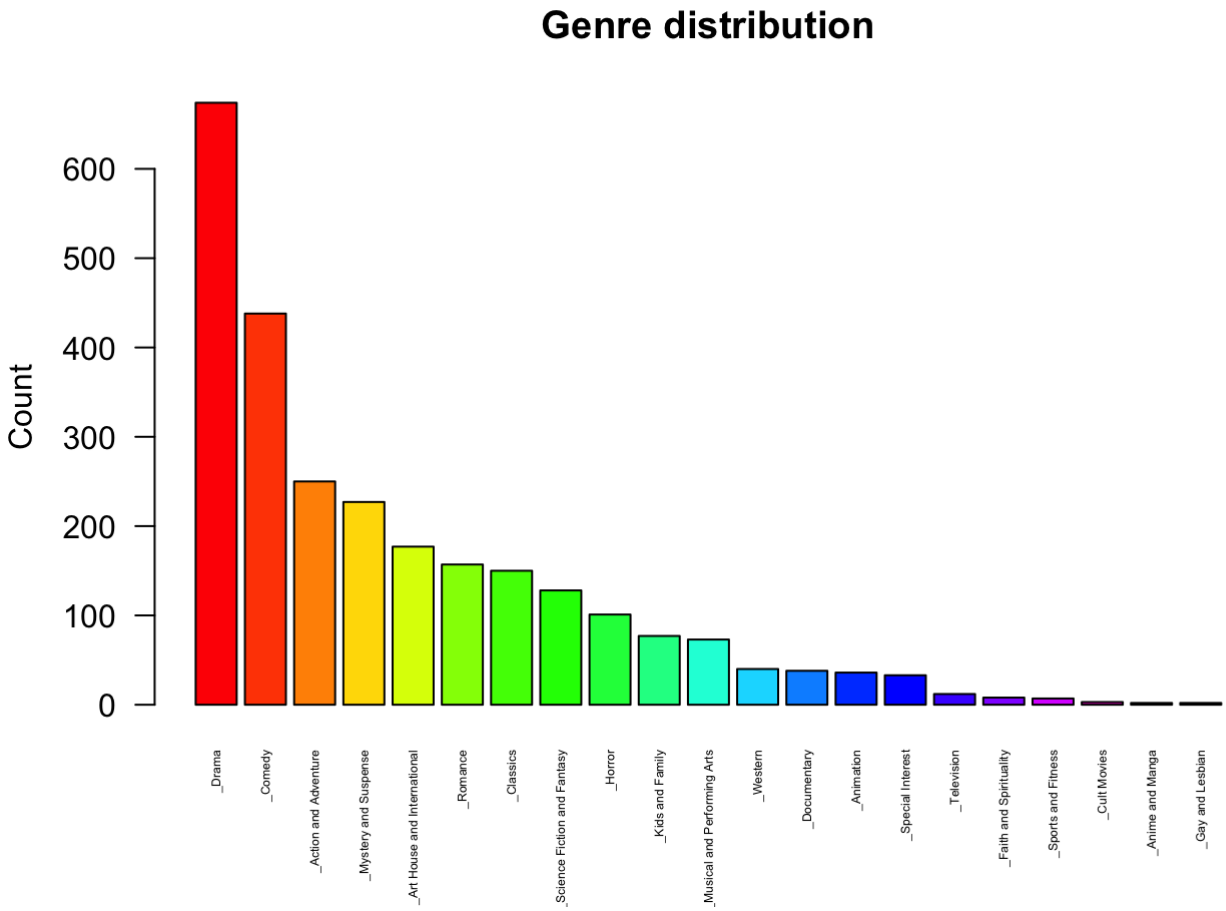

Figure 5 : Number of films for each genre

Regarding the year in which the reviews were written, the first year is 1800 (probably due to mistyping) and the last year is 2018. We have seen that until 2000 there are very few reviews (24 in total), the year with most reviews is 2000 (**RottenTomatoes** was officially launched in 2000) and then from 2000 to 2018 in mean we have around 2720 reviews per year (see *Figure 6*).

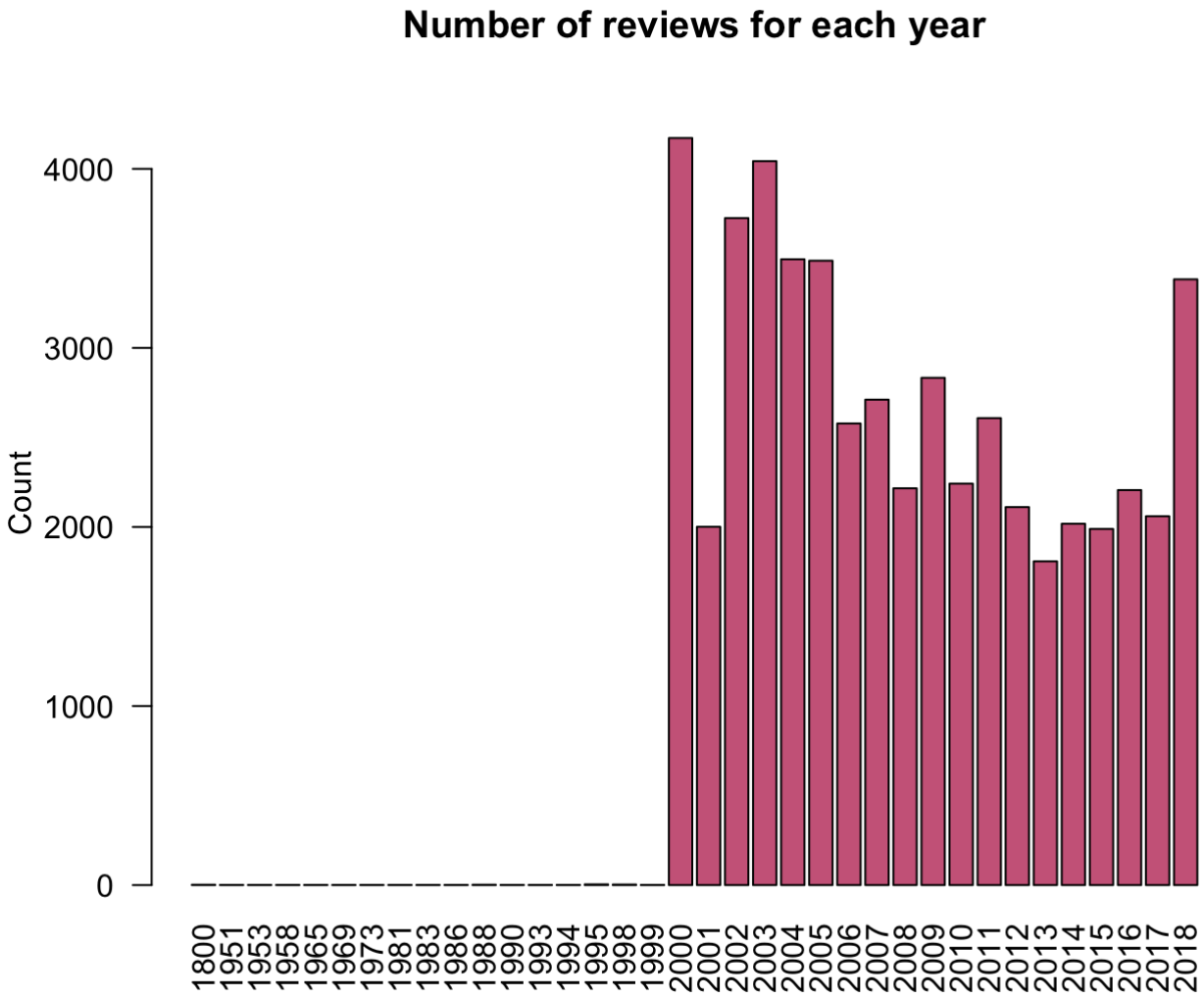

Figure 6 : Number of revires for year

Table creation for the binary approach

To learn **Recommender Systems** using the **R** package **recommenderlab** for the *binary* approach we have created a table that has on the rows the users' name and on the columns the films' id. When there is a correspondence user-film, so when critic in row i made a review of movie in column j in position (i, j) of the table there is the value of this review (1, if the critic evaluated that film as *fresh*, 0 if as *rotten*).
Since some of the critics reviewed the same film more than once, we have deleted in **reviews** the rows that correspond to reviews made by the same critic more than once keeping only the most recent review. The critics that reviewed the same film more than once are 100 and in total we have deleted 285 binary ratings from the original 51710. Most of these critics reviewed only one film two times, there is only one exceptional case, the one of critic Nell Minow that made 92 double reviews.
Figure 7 shows the number of reviews made for each of these critics, while *Figure 8* shows which of the critics tend to change idea by reviewing the same film more times and which ones do not. From *Figure 8* it is possible to see that most of the critics do not change idea (91%), 6% of them change always idea and 3% change idea less than 50% of the times.

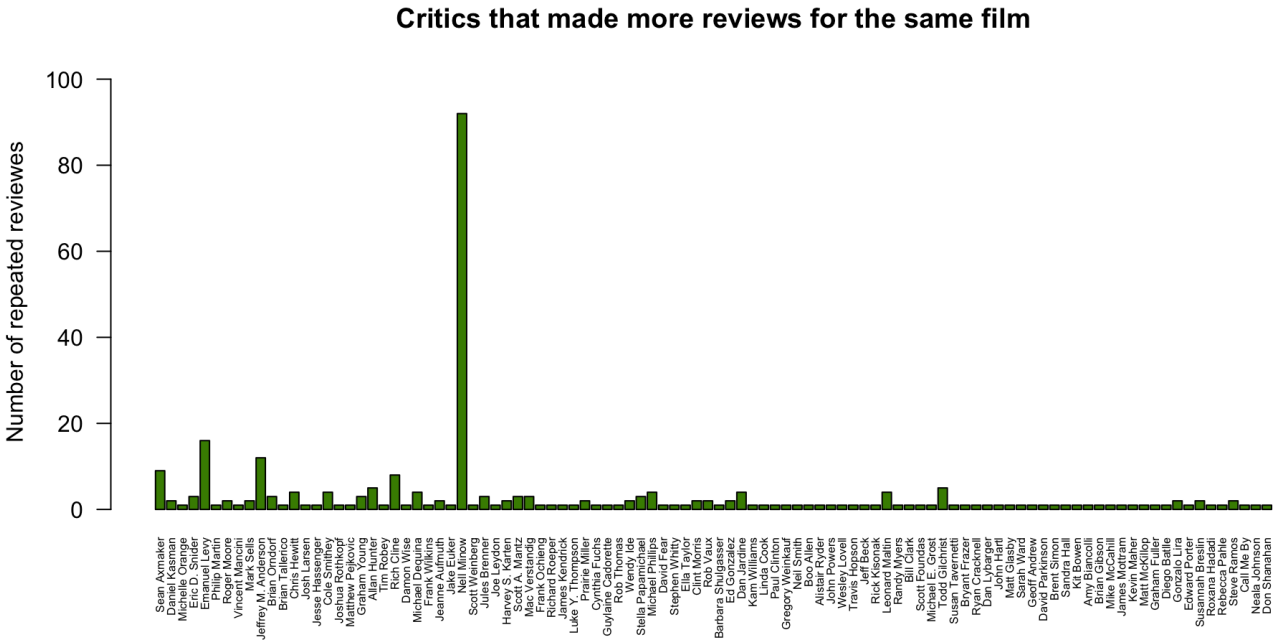

Figure 7 : Number of reviews for the same film by the same user

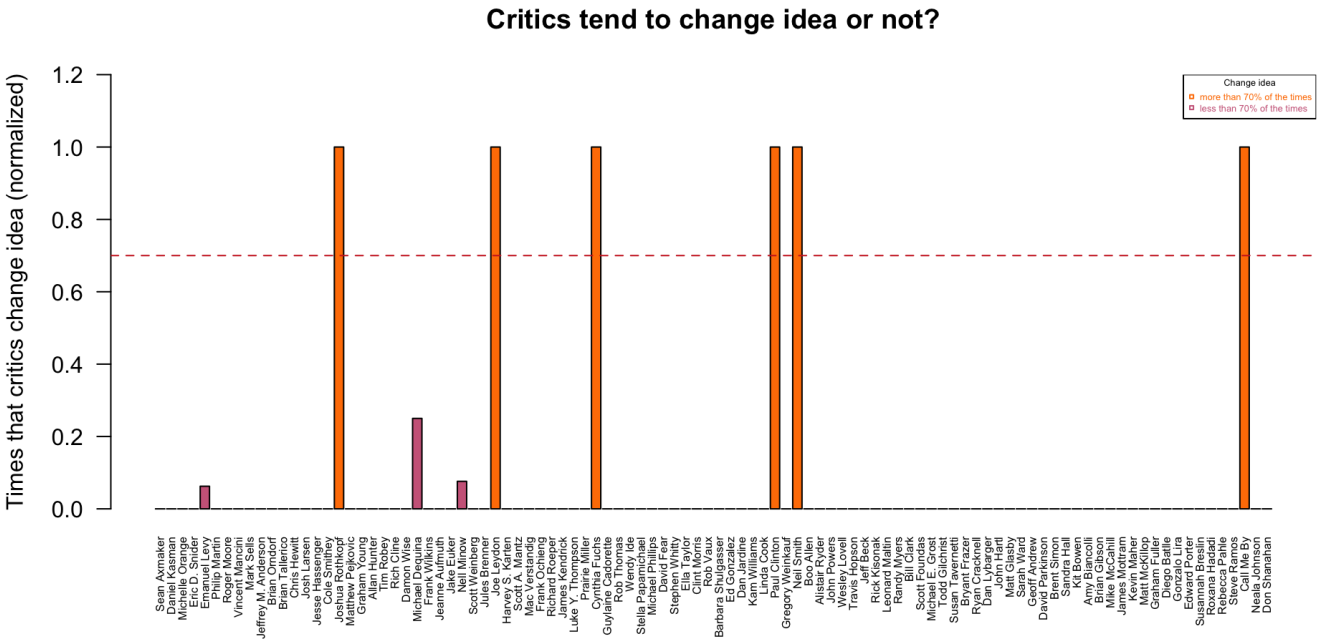

Figure 8 : Percentage of time that critics change idea reviewing the same film more than once

Using **reviews** dataset, with only the most recent reviews for the critics that reviewed the same film more than once, we have obtained **bin** table (table for the *binary approach*) with 3496 critics and 1133 films.

Improvement: Active user after 2010

We have improved our approach taking into account the date when the review was written and thus using only effective active users. We have considered *active users* those who wrote their last review after 2010. By doing so, we have obtained 2420 active users and a total of 18115 reviews. The new table, **binAU**, was obtained keeping from **bin** only the rows that correspond to active users.

Table creation for the non-binary approach

Let **nonbin** be the table for the *non-binary* approach, it contains the non-binary ratings that critics gave to the movies reviewed by them. **nonbin** was created with 3849 critics and 1132 movies.
As for the *binary* approach, we have removed from **reviewsNONBIN** the oldest reviews made by the same

critic for the same film. These critics are the same as before and in total we have erased 283 reviews from the original 51654. *Figure 9* shows for these critics the number of times that they have reviewed the same film more than once and *Figure 10* shows the percentage of the occasions where they have changed the rating for the same film.

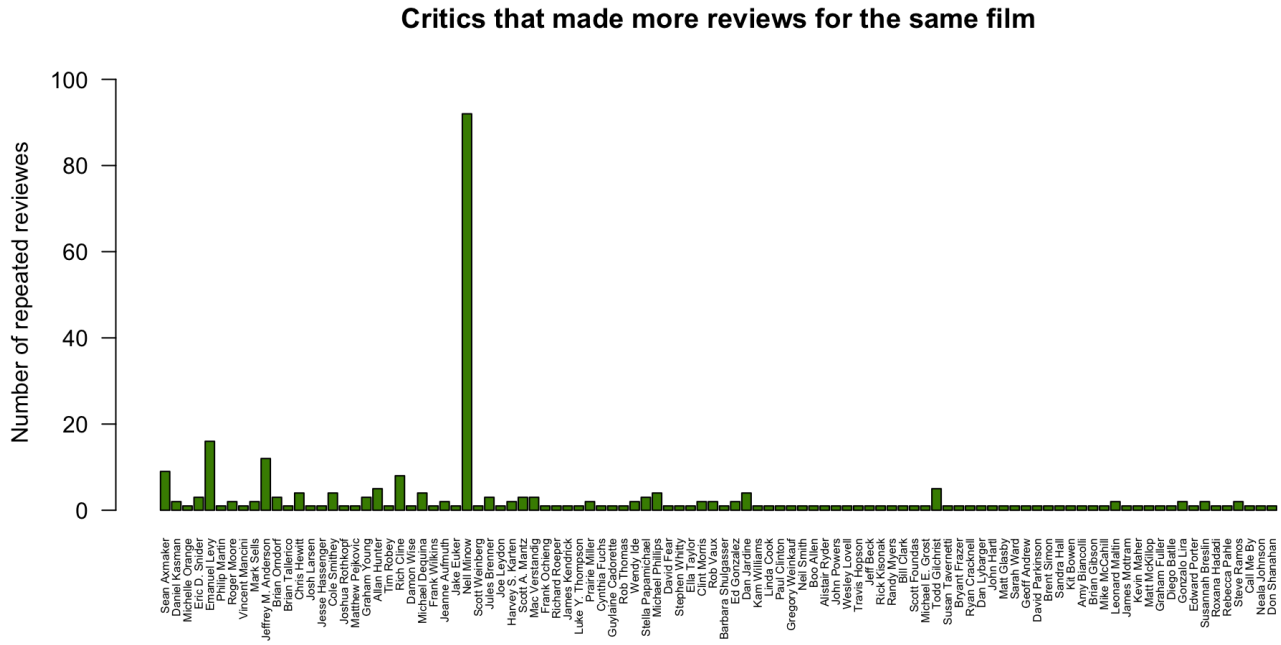

Figure 9 : Number of reviews for the same film by the same user

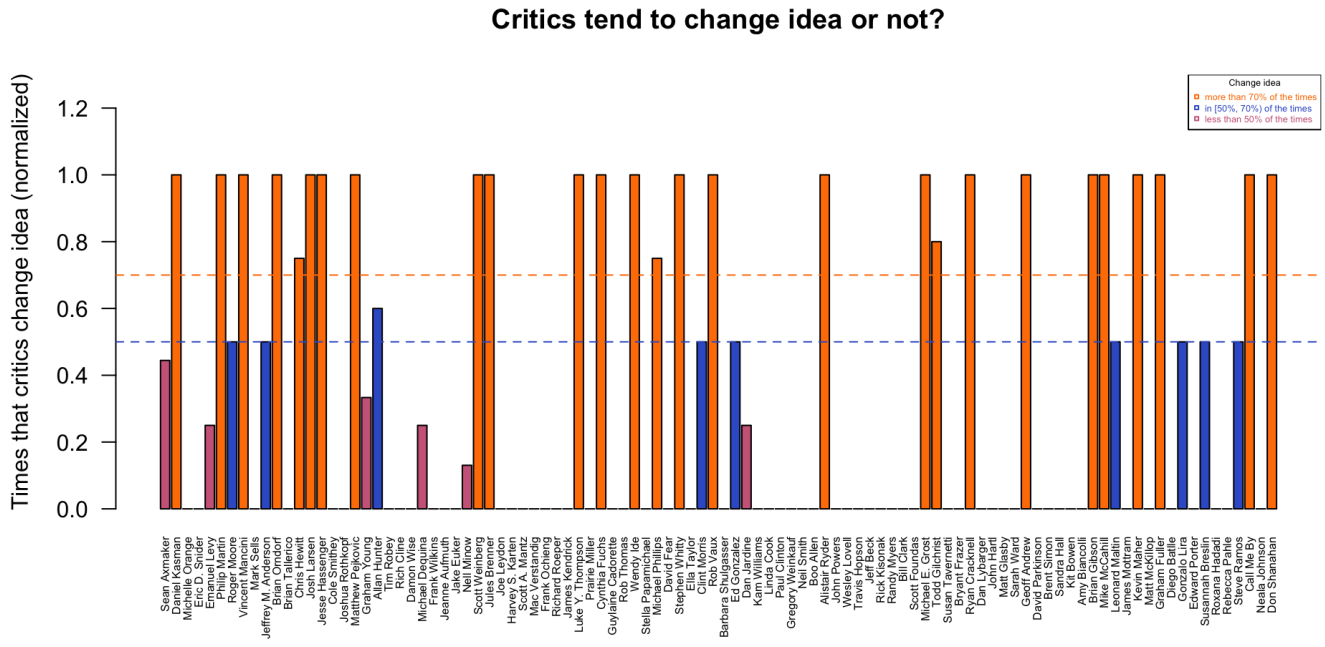

Figure 10 : Percentage of time that critics change idea reviewing the same film more than once

Using **reviewsNONBIN** dataset with the most recent reviews for the critics that reviewed the same film more than once we have obtained **nonbin** (table for *non-binary* approach) with 3489 critics and 1132 films.

Improvement: Active user after 2010

Also for the *non-binary* approach we have learned a reccomender system using only the active users (active after 2010). In this case the active users are 2413 and they have done 18093 reviews. The new table, **nonbinAU**, was obtained keeping from the **nonbin** table file only the lines that correspond to the active users.

Task 2- Recommender systems

We have used the reviewers' ratings of movies to learn different recommender systems to **recommend Top-N movies for a given reviewer** by:

- Popularity;
- Association Rules;
- Collaborative Filtering.

Recommender systems is an application of web mining techniques (one of the area of study in data mining), the goal of these techniques is to help user in the research of information on the web based on previously recorded data. This is actually the purpose of our work, we want to help users to find movies that they probably will like basing the recommendation on the films and reviews made by each user (information that we have thanks to the website **RottenTomatoes**). This can be useful not only for users who are speeded up in their searches but also for the site that can improve its performance in terms of customer loyality.

Our recommendation problem can be formulated as follows:

- We have a set of users U, the critics;
- We have a set of items I to be recommended to the users, the movies;
- Each user U is defined with a user profile, his reviews and ratings;
- Our goal is to learn the utility function that returns how much an item $i \in I$ could be useful/appreciated by user $u \in U$ and predict a list of movies that each user is likely to see.

There are different approaches to build recommender systems: one is based on the items that the user has liked in the past (this is called **Content-Based** approach), the other is based on the similarity between users and items (this is called **Collaborative filtering** approach). There are also hybrid approaches that consist in combination of the previously mentioned approaches [1].

A simpler but less robust type of recommendation that we have also studied in this project is made through **Popularity**, this consists in suggesting to the users the most popular films (therefore the most seen ones) eliminating from this list those already seen by the user in consideration.

We have also made recommendation using **Association Rules** (*Content-Based* approach). This algorithm works as follows:

1. An active user saw the list of movies M;
2. The algorithm looks for rules $X \rightarrow Y$ (where $X \subseteq M$);
3. Then the items in Y that are already in M are delated from Y (we do not want to suggest movies already seen by the critic);
4. The movies in X are sorted by decreasing confidence ($confidence(X \rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$) and, in case of same confidence, the one with higher support ($support(X \rightarrow Y) = supp(X \cup Y)$) are preferred;
5. Y is the list of movies that can be suggested to the critics.

To do this we have had to use a low support (we choose 0.01) because of the sparsity of the data. In our model we have set a confidence equal to 0.5 in order to detect as recommended movies the relevant ones.

The last algorithms for recommendation that we have used in this project are based on **collaborative filtering (CF)**, these algorithms are typically divided into two groups, *memory-based CF* and *model-based CF* algorithms. The first one uses the whole database to create recommendations, this approach can leads to problems in scalability since the whole user database has to be processed online for creating recommendations. Model-based algorithms uses the user database to learn a more compact model [1]. We have analyzed two different methods of collaborative filtering based on these different groups:

- UBCF, **User-Based Collaborative Filtering**, which is based on the fact that similar users have similar ratings on the same item (*memory-based CF*);
- IBCF, **Item-Based Collaborative Filtering**, which is based on the fact that similar items are rated in similar way by the same user (*model-based CF*).

There are different ways to define the similarity between items/users, as *cosine similarity*, *Jaccard similarity*, etc. In the model done by us we have used *cosine similarity* whose formulas are:

- user-based: $sim(u,v) = \cos(\overline{u}, \overline{v}) = \frac{\sum_{i \in I} u_i v_i}{\sqrt{\sum_{i \in I} u_i^2} \sqrt{\sum_{i \in I} v_i^2}}$
- item-based: $sim(i,j) = \cos(\bar{i}, \bar{j}) = \frac{\sum_{u \in U} i_u j_u}{\sqrt{\sum_{u \in U} i_u^2} \sqrt{\sum_{u \in U} j_u^2}}$

Having these informations on the similarity between users and items it is possible to produce the **UBCF** and **IBCF**. Below are the steps to build recommendations for the *binary* and the *non-binary* approaches.
The **user-based** reccomendations are produced according to the following steps:

1. Given an active critic, u;
2. Set $N(u,k)$ the k nearest neighobors of u;
3. Predict the score that critic u will give to each movies based on its neighbours. Let $\hat{r}_{ui}$ be the score that the critic is going to give to movie i.

   i. In the *binary* case it is: $\hat{r}_{ui} = \sum_{v \in N(u,k)} sim(u,v) viewed(v,i)$, where $viewed(v,i)$ is 1 if the neighbor v of u has seen movies i, 0 otherwise.
   ii. In the *non-binary* case there are different ways to calculate it (weighted, unweighted, weighted and mean centered), we have used the weighted and mean centered method (default one) that is: $\hat{r}_{ui} = \overline{r}_u + \frac{\sum_{v \in N_i(u,k)} w_{uv}(r_{vi} - \overline{r}_v)}{\sum_{v \in N_i(u,k)} w_{uv}}$, where $N_i(u,k)$ are the k-nearest neighbors of user u who have rated item i, w_{uv} is the weight between critics u and v (their similarity) and $\overline{r}_u$ is the mean of the ratings made by user u.

4. Recommend the items with highest score.

The steps for the **item-based** reccomendations are:

1. Given an active session s_a (set of movies seen by one user);
2. Compute the score for each movies i doing:

- find the k nearest neighbors of i, $N(i,k)$;
- compute $\hat{r}_{(s_a,i)} = \frac{\sum_{j \in s_a \cap N(i,k)} sim(i,j)}{\sum_{j \in N(i,k)} sim(i,j)}$ for the *binary* case, and $\hat{r}_{ui} = \overline{r}_i + \frac{\sum_{j \in N_u(i,k)} w_{ij}(r_{uj} - \overline{r}_j)}{\sum_{j \in N_u(i,k)} w_{ij}}$ ($N_u(i,k)$ are the k-nearest neighbors of movies i rated by critic u) for the *non-binary* case;

3. Reccomend movies with highest score.

We have build recommendation model using the **R** package **recommenderlab**.

So what we have to do to recommend the Top-N movies to an active user u_a is, after having predicted the ratings of the film not viewed by him, to take from this list the N items with the highest prediting ratings. It could happens that the number of items that user u_a has not seen has cardinality less then N or that the algorithm is not able to identify N movies to recommend, so in these cases the list of Top-N movies wuold have less than N elements.

After having constructed the recommender system we have to **evaluate** its goodness. The first step to do consists in divide the table of the ratings partitioning the user as U_{train} and U_{test}, the first set is used to learn the recommender system while the second one to evaluate its goodness.
There are different parameters that tell how good is an evaluation, like predictive power, novelty, serendipity, diveristy, robustness, privacy perserving, etc.

Table 2 could be useful to facilitate the interpretation of the parameters that we have used.

	Recommended	***Not recommended***
Preferred	True-Positive (tp)	False-Negative (fn)
Not preferred	False-Positive (fp)	True-Negative (tn)

Table 2: Classification of the possible result of a recommendation of an item to a user [2].

Some metrics that are good to evaluate the recommendations done are:

- The **recall** that is defined as the ratio of relevant item guessed, $Recall = Sensitivity = \frac{\#tp}{(\#tp+\#fn)}$. So is the ratio between the correct recommendation and the totality of the possible useful recommendation.
- The **precision** represents the quality of each recommendation, $Precision = \frac{\#tp}{(\#tp+\#fp)}$. So it is the ratio between the correctly recommended movies over the total number of movies recommended, $i.e.$ precision is the percentage of movies indeed enjoyed from all the movies recommended to the user.

Ideally **precision** and **recall** should be high: we want the model to give us a high percentage of recommendations that we will indeed enjoy and we also want that the model recommends us only our preferred ones!
Precision and recall have confliting properties, since high precition means low recall and vice-versa [1].

In the case of the *non-binary* approach there are other useful values to evaluate the predicted ratings that we have studied that are:

- RMSE, **Root Mean Squared Error**, $RMSE = \sqrt{\frac{1}{|T|}\sum_{(u,i)\in T}(\hat{r}_{ui} - r_{ui})^2}$
- MAE, **Mean Absolute error**, $MAE = \frac{1}{|T|}\sum_{(u,i)\in T}|\hat{r}_{ui} - r_{ui}|$

Where, T is the set of all the pairs (user, item) for which we have a predited ratings $\hat{r}_{ui}$ and a known ratings r_{ui} which was not used to learn the recommender system. Both evaluate the deviation of the prediction from the true value.

We have also used **precision-recall** and **ROC** curves to infer the best recommender system. **ROC** and **precision-recall curves**, both, measure the proportion of preferred items that are actually recommended, but precision-recall curves emphasize the proportion of recommended items that are preferred while ROC curves emphasize the proportion of items that are not preferred that end up being recommended [2].

Binary Ratings

Binary approach with all the users

In the following section we will consider the *binary ratings* that we have obtained from the table **bin.csv**. Our aim is to recommend for a given reviewer Top-N movies (with N = 1, 2, 5) via recommender systems:

The number of ratings in our dataset is 51425 and the maximum number of reviews done by a reviewer is 579, even if, as we have seen in the exploratory analysis, only few reviewers made more than 200 reviews.

We want our model to be as reliable as possible, for this reason we would like to take into account only reviewers that made a sufficient number of reviews. Since we have to keep a suffcient number of reviewers and there are already many rieviewers that made less than 5 reviews (in total 2194), to not have a dataset too poor in terms of users we take into account usual consumers by deleting these 2194 reviewers.

Our new variable *ratings* will contain only 1302 reviewers for a total of 47304 ratings. *Figure 11* shows an histogram that represents the new frequency of the number of *ratings* in the range [5, 70].

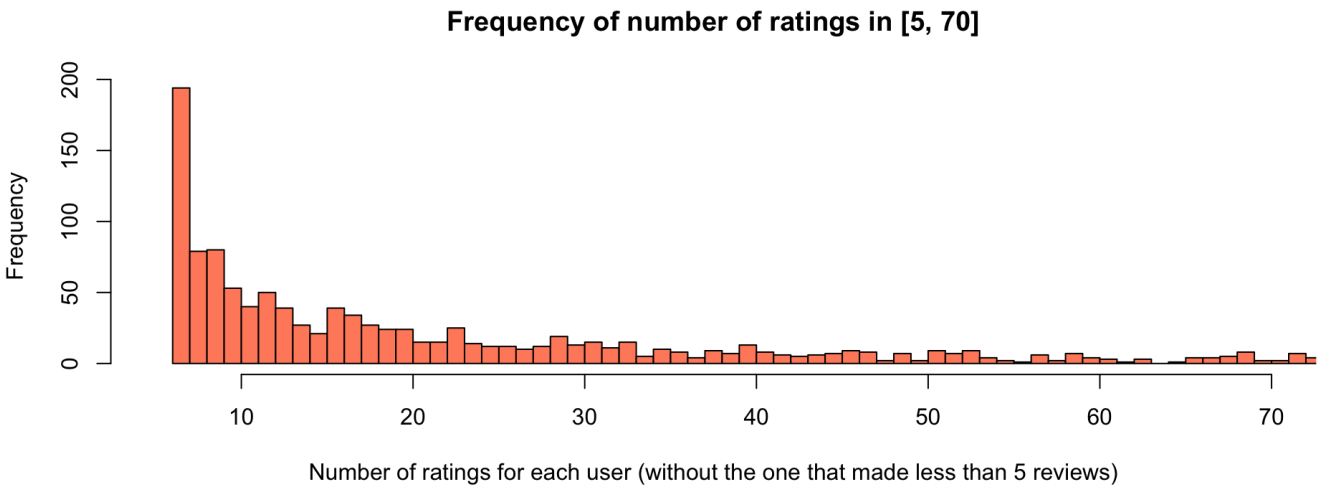

Figure 11 : Frequency of number of ratings in the interval [5 , 70]

We now start to give recommendations through our recommender systems. First of all, we have divided our *ratings* into two parts: *train* and *test*; this is crucial if we want to give recommendations using the **R**-commands **Recommend** and **predict**.
We considered for testing the reccomendation always the same 4 users, chosen basing on their middle average reviews frequency, to be able to compare the different results. Once some critics reviewed ca. 600 movies we have chosen critics that have evaluated half of that movies. Those 4 are: Rich Cline, Nell Minow, Scott Weinberg and Ken Hanke.

Popularity

We recommend Top-5, for the 4 users using the **Popularity** method, this is done by adding in **Recommender** command the method **"Popular"**.

Below the Top-5 recommended movies for test critics using **Popularity**:

```
## $Rich.Cline
## [1] "m1525" "m1083" "m338"  "m1494" "m782"
##
## $Nell.Minow
## [1] "m1646" "m1381" "m1307" "m1935" "m1722"
##
## $Scott.Weinberg
## [1] "m1083" "m1704" "m1017" "m1646" "m841"
##
## $Ken.Hanke
## [1] "m1071" "m1494" "m1960" "m1442" "m782"
```

Association Rules

As we did in the Popularity case, we recommend Top-5 movies for our test users using a recommendation with **Association Rules**. This is done using the method **"AR"** in **Recommender**. The levels of support and confidence used are respectively 0.01 and 0.5, they cannot be too high for the sparsity of our data.

Below the Top-5 recommended movies for test critics using **Associarion Rules**:

```
## $Rich.Cline
## [1] "m1608" "m1758" "m1808" "m782"  "m1944"
##
## $Nell.Minow
## [1] "m1608" "m1758" "m1704" "m1325" "m1960"
##
## $Scott.Weinberg
## [1] "m1608" "m1758" "m1808" "m1017" "m1992"
##
## $Ken.Hanke
## [1] "m65"   "m1071" "m1777" "m1677" "m443"
```

Collaborative Filtering

Now we consider recommendations done with **Collaborative Filtering** and in particular:

- User-Based Collaborative Filtering (UBCF);
- Item-Based Collaborative Filtering (IBCF).

Below the Top-5 recommended movies for test critics using **User-based CF**:

```
## $Rich.Cline
## [1] "m338"  "m1494" "m1517" "m1083" "m1424"
##
## $Nell.Minow
## [1] "m1722" "m528"  "m1857" "m1935" "m1381"
##
## $Scott.Weinberg
## [1] "m976"  "m1585" "m841"  "m1442" "m218"
##
## $Ken.Hanke
## [1] "m1071" "m1009" "m1968" "m1494" "m3"
```

And the Top-5 recommended movies for test critics using **Item-based CF**:

```
## $Rich.Cline
## [1] "m889"  "m222"  "m828"  "m933"  "m1242"
##
## $Nell.Minow
## [1] "m312"  "m824"  "m1133" "m1238" "m1547"
##
## $Scott.Weinberg
## [1] "m28"  "m101" "m163" "m187" "m213"
##
## $Ken.Hanke
## [1] "m28"  "m101" "m163" "m187" "m280"
```

It is possible to observe that different recommender systems recommends different movies to the same user, but sometimes it happens that one of the Top-5 movies recommended is in common between the models, even if the order of importance is not the same, it happens for example with the reviewer Rich Cline and movies *m782* and *m1083*.
We have also observed that for some methods (in particular **Association Rules** and **IBCF**) many movies are in common between the users, specifically it happens for Nell Minow and Scott Weinberg with **AR** and for Scott Weinberg and Ken Hanke with **IBCF**.

Evaluation

After having done these steps, our objective now is to evaluate the recommendations we gave to the test users.

Here below there are the plots regarding to the methods **Popularity** and **Collaborative Filtering** (**IBCF** and **UBCF**).

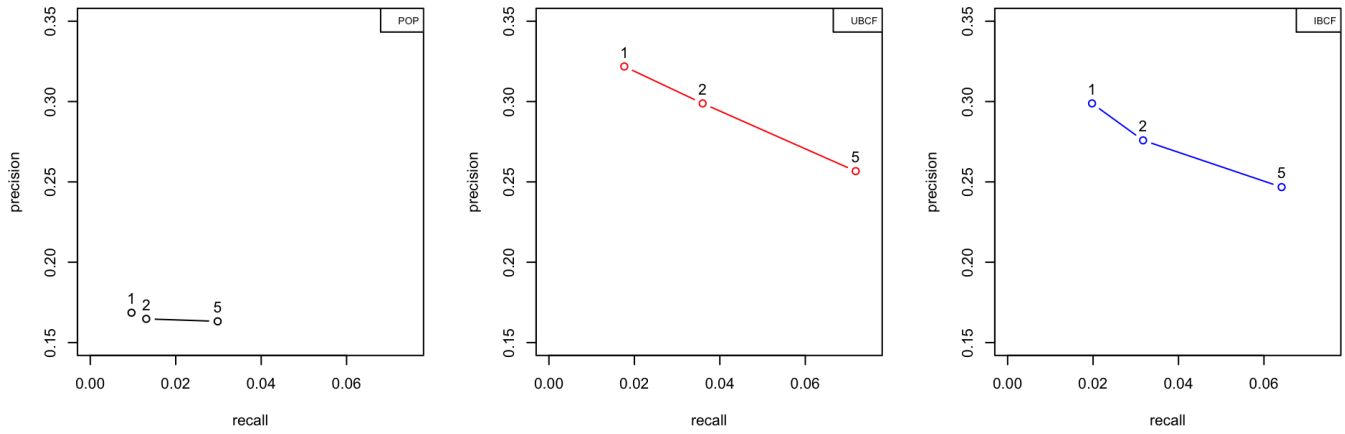

Figure 12 : Recall/Precision for Recommender system methods

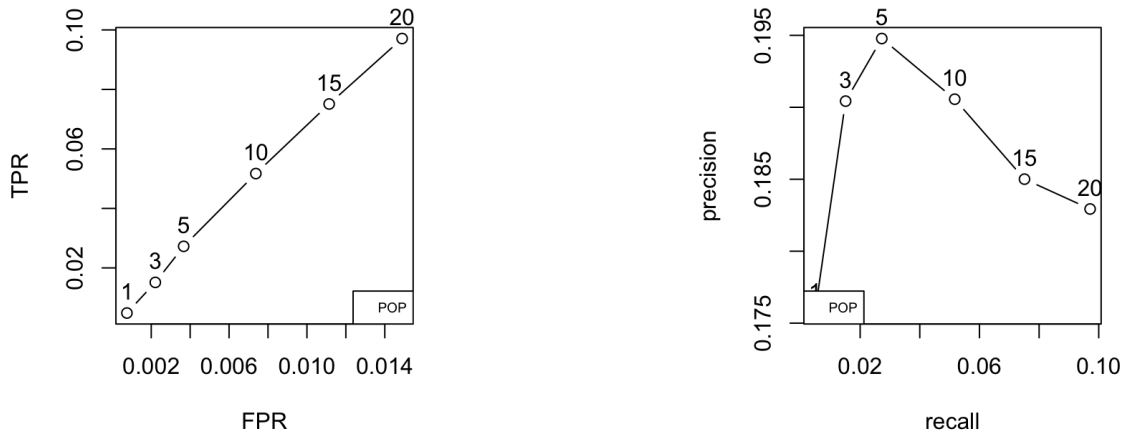

Figure 13 : ROC and precision/recall curves

We have observed (*Figure 13*) that considering many recommendations (top 20) for **Popularity** method, the recall increases while the precision decreases.
That is probably because increasing the number of movies recommended the method becomes less robust in terms of percentage of movies enjoyed from the movies recommended, in fact Popularity method is not specific for a user, it just recommends best movies in general it does not imply that are the best also for the specific user that we are considering.

Active users binary recommender

In order to made a model that takes into account the most relevant reviewers we have decided only to keep the reviewers that did their last evaluation after the year 2010.

In this case we have worked on 2420 reviewers, 1133 movies for a total of 18115 reviews, the maximum number of reviews made by one user is 125.

As done before we decide to delete users that did few reviews, in this case we cancel the ones who did less than 3 reviews. In this way our the numeber of reviewers decreases to 1397. In *Figure 14* it is possible to see the frequency of these reviews.

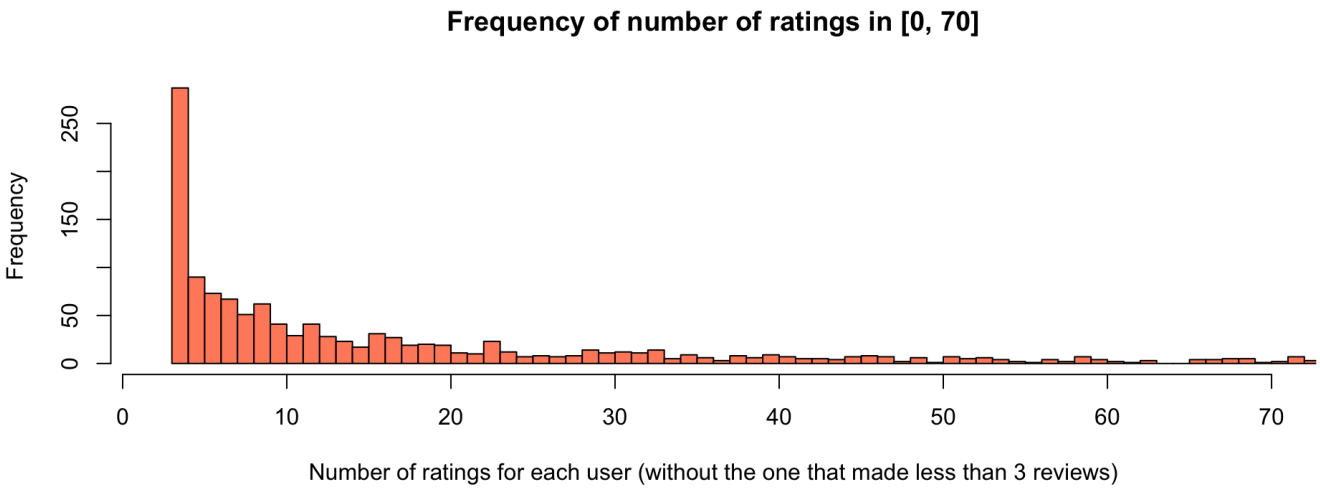

Figure 14 : Number of ratings for each abitual active user

Coherently, we have performed the recommendation step with the usual 4 binary methods.

Here we have Top-5 recommendations for **Popularity** method:

```
## $Rich.Cline
## [1] "X610" "X869" "X438" "X190" "X848"
##
## $Nell.Minow
## [1] "X930" "X599" "X736" "X182" "X781"
##
## $Scott.Weinberg
## [1] "X139"  "X610"  "X966"  "X438"  "X1108"
##
## $Ken.Hanke
## [1] "X139"  "X438"  "X1108" "X848"  "X118"
```

Hereafter we consider the recommendations for **Association Rules** method:

```
## $Rich.Cline
## [1] "X908"  "X994"  "X1023" "X848"  "X438"
##
## $Nell.Minow
## [1] "X908"  "X994"  "X966"  "X748"  "X1088"
##
## $Scott.Weinberg
## [1] "X908"  "X994"  "X1023" "X1088" "X728"
##
## $Ken.Hanke
## [1] "X1088" "X728"  "X347"  "X1121" "X190"
```

Finally, here we present recommendations for Collaborative Filtering (**UBCF** and **IBCF** methods respectively):

```
## $Rich.Cline
## [1] "X190" "X848" "X863" "X610" "X811"
##
## $Nell.Minow
## [1] "X971"  "X182"  "X298"  "X781"  "X1050"
##
## $Scott.Weinberg
## [1] "X553" "X896" "X124" "X476" "X817"
##
## $Ken.Hanke
## [1] "X602"  "X569"  "X1112" "X848"  "X1"
```

```
## $Rich.Cline
## [1] "X504" "X127" "X204" "X324" "X596"
##
## $Nell.Minow
## [1] "X99"  "X324" "X585" "X596" "X86"
##
## $Scott.Weinberg
## [1] "X17" "X56" "X60" "X74" "X87"
##
## $Ken.Hanke
## [1] "X17" "X56" "X60" "X74" "X87"
```

It is possible to observe again what we have seen previously; there are some analogies between the 2^{nd} and the 3^{rd} reviewer in **Association Rules** and between the 3^{rd} and the 4^{th} reviewer in **IBCF**.

It can be observed that recommendations using the same system with active users are very different with respect to the ones done with the original data, in fact on average just one (maximum two) movies are present in both the recommendations.

Then, we have **evaluated** the recommendations:

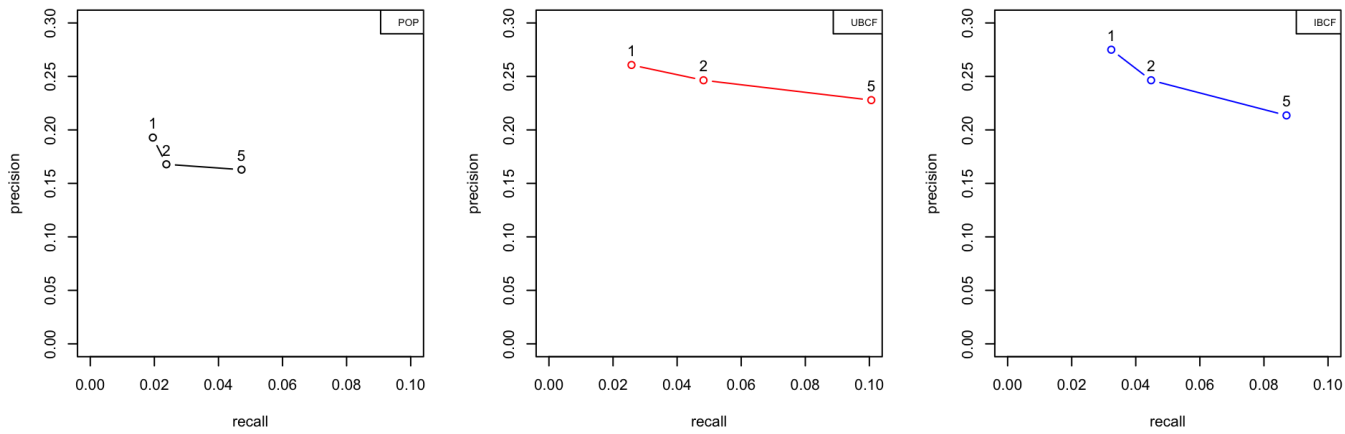

Figure 15 : Recall/Precision for Recommender system methods (Active Users)

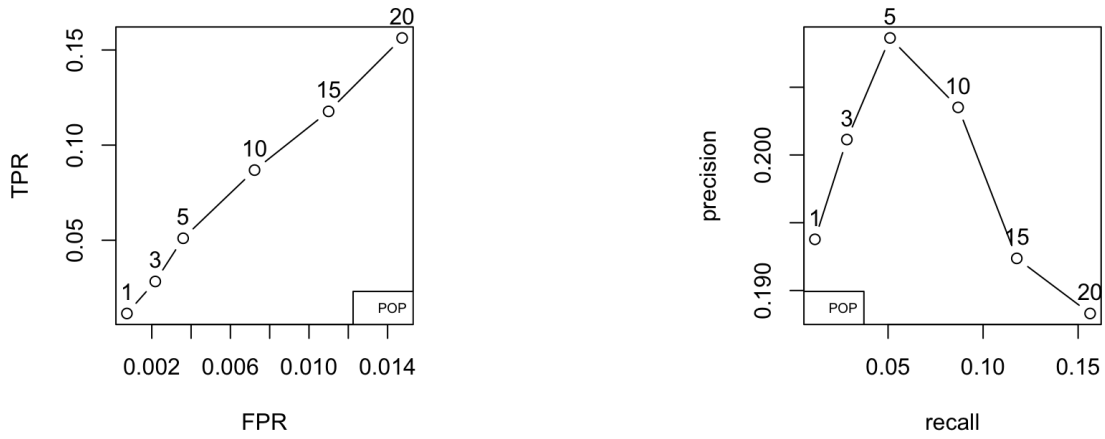

Figure 16 : ROC and precision/recall curves (Active Users)

Again we can observe that increasing the number of recommendations given to the user, the precision of our **Popularity** method decreases, the maximal value for the precision can be reached with 5 recommendations. The behaviour of the curve after this maximum point decreases almost linearly.

Non-binary ratings

Non-binary approach with all the users

We have a dataset with *non-binary ratings* composed by 3489 reviewers and 1132 movies.

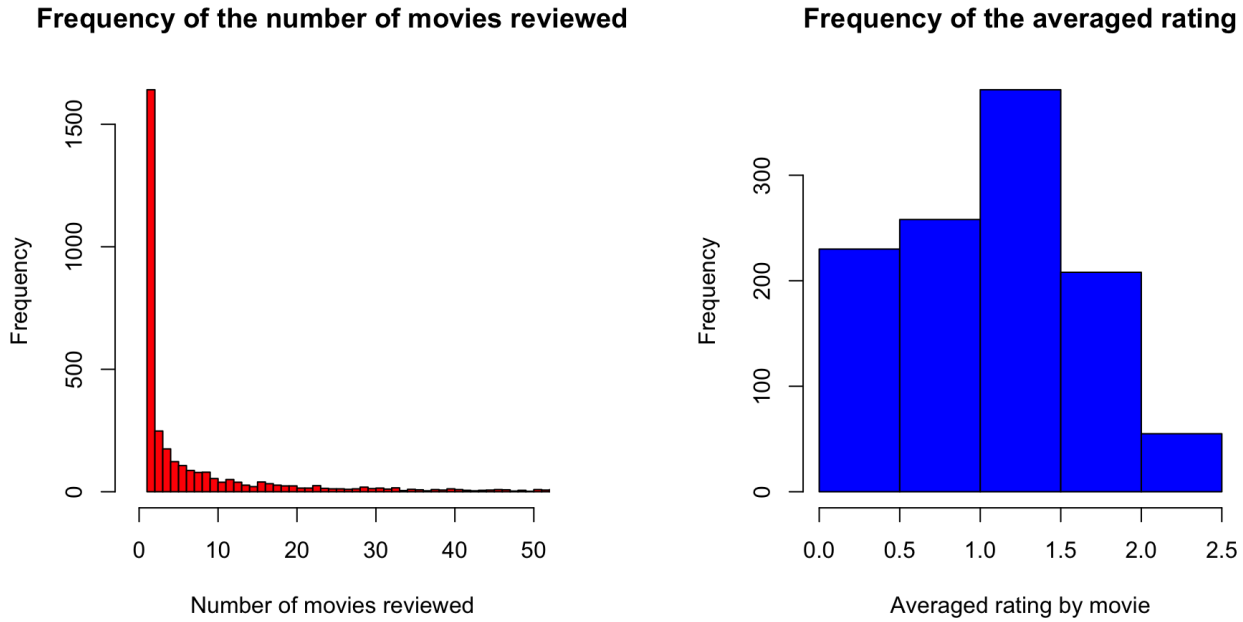

Figure 17 : Frequency of the number of movies reviewed (left). Distribution of the ratings in the non-binary dataset (right).

Figure 17 (left) shows the frequency of the number of movies reviewed. Several critics reviewed less than 6 movies, again those critics were not used for model evaluation in order to improve prediction power of the models. *Figure 17 (right)* also shows the averaged rating per movie.

Following we test the different recommender systems for non-binary ratings: **Popularity**, **User-based Collborative Filtering** and **Item-based Coolaborative Filtering**. **Association Rules** does not apply to non-binary ratings.

Popularity

The output below shows the Top-5 movies recommended to 4 chosen critics, Rich Cline, Nell Minow, Scott Weinberg and Ken Hanke (the same of the *binary* approach). All the *non-binary* recommender system are compared for these 4 critics.

```
## $Rich.Cline
## [1] "m782"  "m1525" "m1494" "m1523" "m582"
##
## $Nell.Minow
## [1] "m1381" "m1067" "m769"  "m528"  "m1788"
##
## $Scott.Weinberg
## [1] "m1960" "m782"  "m57"   "m1381" "m1704"
##
## $Ken.Hanke
## [1] "m1960" "m782"  "m1067" "m1494" "m1523"
```

The output below shows the predicted ratings for the Top-5 recommended movies using the **Popularity** method for the critic Nell Minow.

```
##    m1381    m1067     m769     m674    m1890
## 2.435193 2.377036 2.367250 2.141231 2.030959
```

User-based and Item-based Collaborative Filtering

The same methodology was used for the **UBCF** and **IBCF** methods:

```
## $Rich.Cline
## [1] "m334"  "m17"   "m1144" "m26"   "m33"
##
## $Nell.Minow
## [1] "m54"   "m1775" "m304"  "m582"  "m119"
##
## $Scott.Weinberg
## [1] "m3"    "m17"   "m1704" "m356"  "m1144"
##
## $Ken.Hanke
## [1] "m1418" "m1890" "m165"  "m1803" "m1656"
```

```
##     m582     m583     m986    m1992    m1163
## 1.771190 1.717857 1.748968 1.764524 1.717857
```

```
## $Rich.Cline
## [1] "m8"   "m52"  "m79"  "m94"  "m542"
##
## $Nell.Minow
## [1] "m1155" "m968"  "m54"   "m432"  "m1055"
##
## $Scott.Weinberg
## [1] "m805"  "m1313" "m724"  "m1410" "m1821"
##
## $Ken.Hanke
## [1] "m1347" "m1659" "m475"  "m744"  "m617"
```

```
##     m968      m61     m121     m490     m613
## 2.666667       NA 2.000000 1.750000 2.000000
```

Evaluation

Following we have performed model comparison by calculating error and creating the precion/recall curves for evaluating the different models. We have created an evaluation scheme which splits non-binary ratings into a training set (80%) and a test set (20%).

The output bellow shows that the lowest errors in ratings prediction are obtained by the **Popularity** method.

```
##           RMSE       MSE       MAE
## POP  0.8504530 0.7232703 0.6650061
## UBCF 0.9104034 0.8288344 0.7246889
## IBCF 1.0983342 1.2063380 0.8183099
```

Figure 18 shows the precision/recall curves for the 3 methods. Method **UBCF** is the best recommender system for this dataset. *Figure 18* also shows that predicting is very poorer when recommending more than 2 movies with these systems.

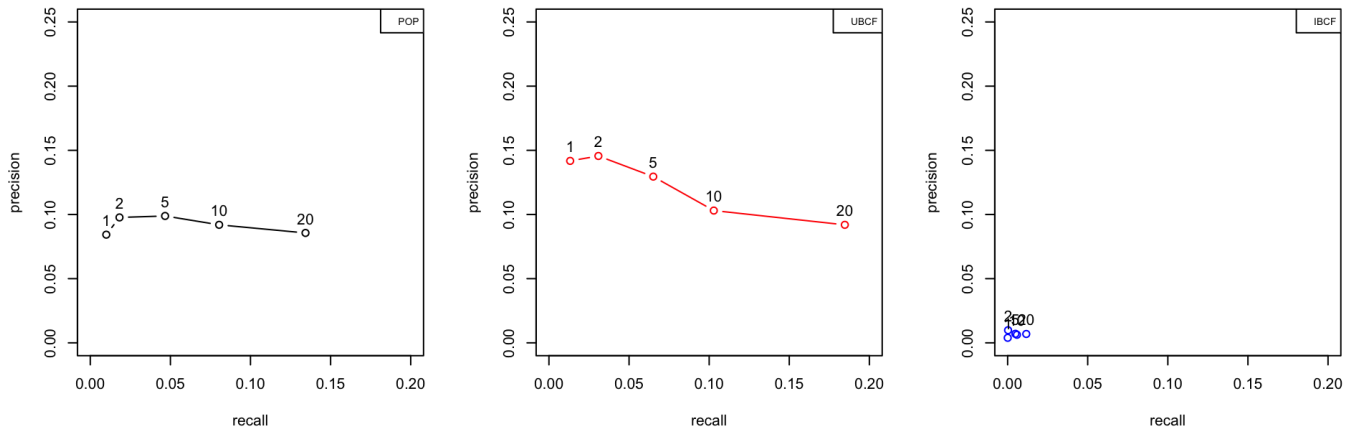

Figure 18: Precision/recall curves.

Active users non-binary recommender

As we have done in the *binary* approach we have considered also the Recommender models with only the active users. In this way we have a dataset with 2413 reviewers and 1132 movies (ca. 30% reduction from the whole non-binary dataset).

The output below shows the recommended movies for the same 4 chosen users for the **Popularity**, **UBCF** and **IBCF** systems, respectively. **Popularity** based recommendations are nearly the same, $i.e.$, date of the last reviews did not influence recommendations (campare with reccomendations above). However, the same does not happen with **UBCF** and **IBCF**.

```
## $Rich.Cline
## [1] "m782"  "m1525" "m1494" "m1523" "m582"
##
## $Nell.Minow
## [1] "m1067" "m1381" "m769"  "m1788" "m582"
##
## $Scott.Weinberg
## [1] "m1960" "m782"  "m57"   "m1704" "m1067"
##
## $Ken.Hanke
## [1] "m1960" "m782"  "m1067" "m1494" "m1523"
```

```
## $Rich.Cline
## [1] "m782"  "m1523" "m26"   "m1431" "m953"
##
## $Nell.Minow
## [1] "m54"   "m1775" "m199"  "m283"  "m1163"
##
## $Scott.Weinberg
## [1] "m3"    "m54"   "m1704" "m769"  "m394"
##
## $Ken.Hanke
## [1] "m1890" "m1399" "m782"  "m394"  "m1067"
```

```
## $Rich.Cline
## [1] "m52"  "m121" "m284" "m466" "m542"
##
## $Nell.Minow
## [1] "m968"  "m432"  "m655"  "m1123" "m977"
##
## $Scott.Weinberg
## [1] "m805"  "m1207" "m1821" "m1657" "m102"
##
## $Ken.Hanke
## [1] "m1347" "m1659" "m475"  "m744"  "m617"
```

Evaluation

Error rates are nearly the same as before and now taking into account only active users.

```
##           RMSE       MSE       MAE
## POP  0.8155504 0.6651224 0.6462769
## UBCF 0.8843681 0.7821069 0.7015407
## IBCF 1.0691435 1.1430678 0.7787611
```

Figure 19 shows the corresponding precision/recall curves. **UBCF** is the best recommender but recommendentaion is poorer for more than 1 movie recommendation.

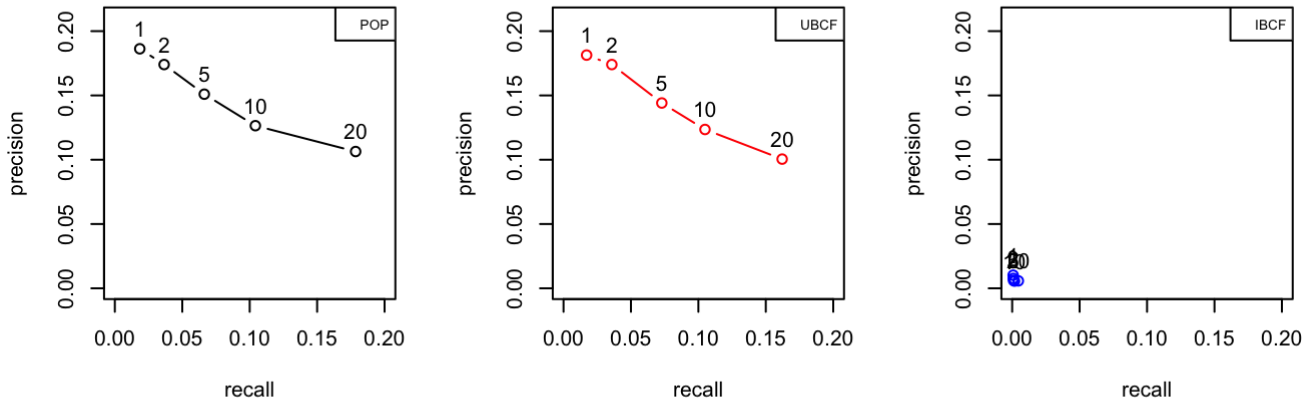

Fig. 19: Precision/recall curves for the active users non-binary dataset.

Task 3 - Context-aware Recommendation

Let's suppose that when we have more free time (usually during weekends) we are happier, our mood influences our choices and our personal grade of satisfaction.
In order to prove our thesis we have decided to split our data set into two parts: **Weekdays** and **Weekend**, based on the day the review was made.
We have repeated all the steps done previously for these two subsets of data.

Binary context-based approach

The number of ratings during the Weekdays is higher than the one relative to Weekends. For this reason we have decided to keep all the ratings for the *Weekend* and erase the reviewers that did less than 5 reviews in the *Weekday* dataset.
We start from the **Weekday**.

Weekday

Here reported, we show the frequency of the ratings after cleaning the "lazy" reviewers (we only kept reviewers that made more than 5 reviews in order to have a trustworthy data set).
Here we present the histogram relative to the frequency of the weekday reliable users.

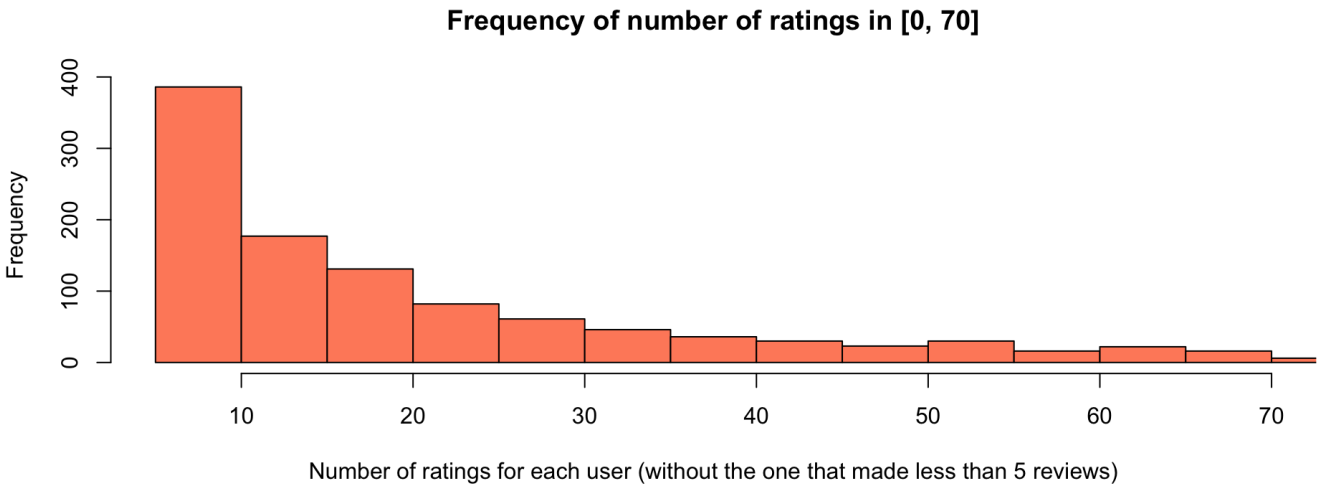

Figure 20 : Frequency of ratings of weekday users

Here we present the results obtained with our usual methods: **Popularity**:

```
## $Rich.Cline
## [1] "m782"  "m1083" "m1525" "m1494" "m1523"
##
## $Nell.Minow
## [1] "m1646" "m1381" "m77"   "m1935" "m322"
##
## $Scott.Weinberg
## [1] "m782"  "m1083" "m1960" "m841"  "m251"
##
## $Ken.Hanke
## [1] "m782"  "m1071" "m1960" "m251"  "m1494"
```

Association Rules:

```
## $Rich.Cline
## [1] "m1608" "m1758" "m338"  "m1183" "m1777"
##
## $Nell.Minow
## [1] "m1608" "m1758" "m1325" "m1857" "m1376"
##
## $Scott.Weinberg
## [1] "m841"  "m443"  "m1844" "m1585" "m1685"
##
## $Ken.Hanke
## [1] "m1071" "m841"  "m1475" "m1588" "m218"
```

User-based Collaborative Filtering:

```
## $Rich.Cline
## [1] "m338"  "m1496" "m1588" "m528"  "m1494"
##
## $Nell.Minow
## [1] "m1223" "m1968" "m1962" "m3"    "m1935"
##
## $Scott.Weinberg
## [1] "m841"  "m1585" "m1223" "m976"  "m862"
##
## $Ken.Hanke
## [1] "m1496" "m1968" "m1009" "m1071" "m3"
```

Item-based Collaborative Filtering:

```
## $Rich.Cline
## [1] "m1242" "m1331" "m1496" "m1363" "m559"
##
## $Nell.Minow
## [1] "m242"  "m1060" "m192"  "m219"  "m1776"
##
## $Scott.Weinberg
## [1] "m94"  "m101" "m163" "m187" "m213"
##
## $Ken.Hanke
## [1] "m94"  "m101" "m163" "m187" "m280"
```

We can observe what we have registered in the binary recommendation done with complete data set or just with active users; only in few cases same movies are recommended to more users, it only happens to the same reviewers for the same Recommender methods and in the majority of the cases the common movies are the ones that were suggested also in the previous sections.

Then, we have done the **evaluation** part for the **Weekday** data-set:

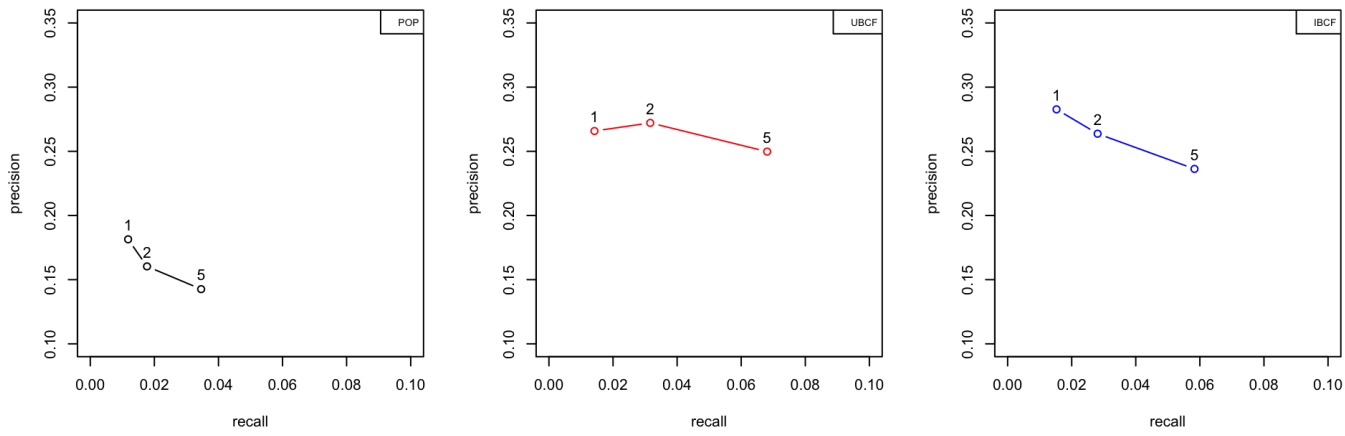

Figure 21 : Recall/Precision for Recommender system methods (Weekdays)

As usual, the behaviour of **Popularity** and **IBCF** are similar, while the **UBCF** method has a maximum precision in corrispondence of Top-2 movies suggested.

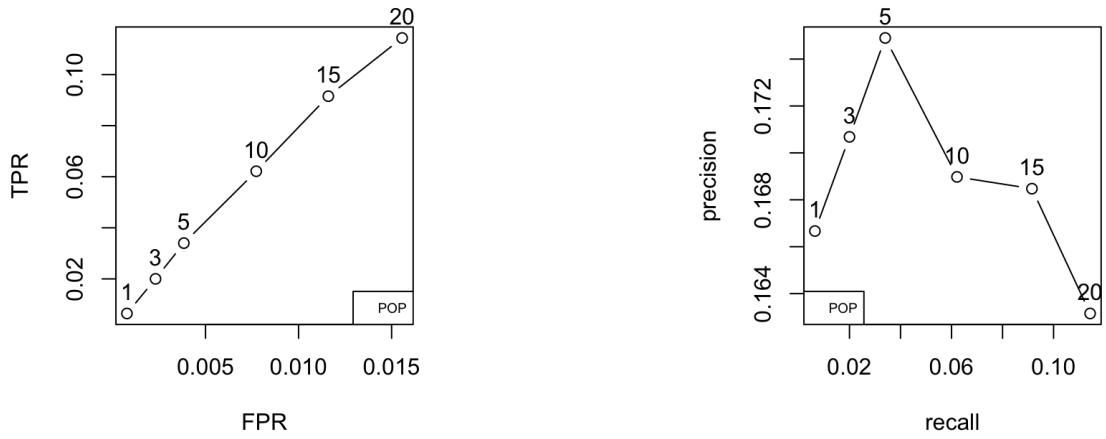

Figure 22 : ROC and precision/recall curves (Weekday Users)

Weekend

As said, since we have less data for the **Weekend** days, we prefer not to delate the “lazy” reviewer in order to grant the biggest data set possible.

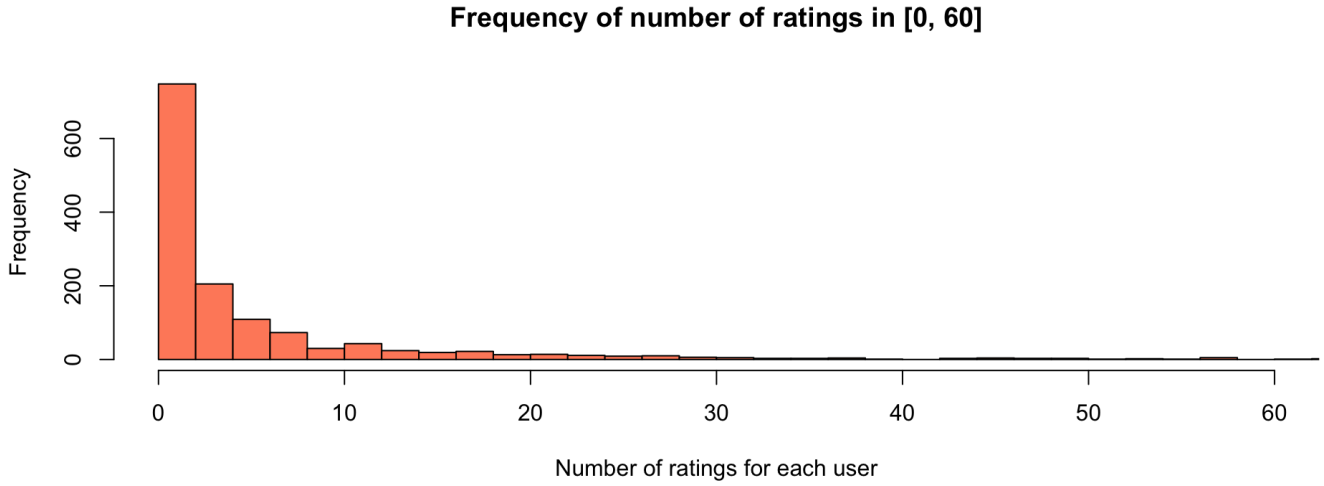

Figure 23 : Frequency of number of ratings during the weekend in the interval [0, 60]

We perform the usual Recommendation using **Popularity**, **Association Rules** and **Collaborative Filtering**.

Here we present the results obtained with our usual methods: **Popularity**:

```
## $Rich.Cline
## [1] "m235"  "m1446" "m1750" "m1418" "m1751"
##
## $Nell.Minow
## [1] "m235"  "m1750" "m1418" "m1767" "m118"
##
## $Scott.Weinberg
## [1] "m1446" "m1751" "m1767" "m370"  "m211"
##
## $Ken.Hanke
## [1] "m235"  "m1446" "m1750" "m1418" "m1751"
```

Association Rules:

```
## $Rich.Cline
## [1] "m1724" "m119"  "m1392" "m1840" "m947"
##
## $Nell.Minow
## [1] "m1800" "m1843" "m235"  "m376"  "m1418"
##
## $Scott.Weinberg
## [1] "m1608" "m1758" "m947"  "m1446" "m1750"
##
## $Ken.Hanke
## [1] "m655"
```

User-based Collaborative Filtering:

```
## $Rich.Cline
## [1] "m1442" "m1935" "m1183" "m301"  "m1877"
##
## $Nell.Minow
## [1] "m235"  "m1843" "m118"  "m1767" "m1800"
##
## $Scott.Weinberg
## [1] "m1767" "m1843" "m1446" "m1881" "m1751"
##
## $Ken.Hanke
## [1] "m809"  "m232"  "m305"  "m1219" "m655"
```

Item-based Collaborative Filtering:

```
## $Rich.Cline
## [1] "m252"  "m1051" "m1054" "m1164" "m1172"
##
## $Nell.Minow
## [1] "m293" "m323" "m457" "m588" "m772"
##
## $Scott.Weinberg
## [1] "m293" "m457" "m502" "m588" "m772"
##
## $Ken.Hanke
## [1] "m20" "m24" "m50" "m51" "m74"
```

We observe that the user Ken Hanke has just one movie suggested with the **Association Rules** method, the reason might be that he did't see enough movies to receve 5 recommendations with the values of confidence and support we have imposed. In fact the mean of the movies reviewed by the other 3 users is 64, and the number of movies reviewed by Ken Hanke is just 23.

The results obtained from **Weekend** analysis and **Weekeday** analysis are completely different, the suggested movies are different, this could mean that the reviews in the two data sets are different, maybe because the genre of movies seen during the weekday are in general different from the ones seen in the weekend, this hypothesis could lead to the thesis that the people mood influences the choice of a movie and consequently its rating, this is not applicable for the *binary* case since the rating is just [0-1]. A social study might, probably, solve this dilemma; that the mood influences the selection of a product (in our case a movie).

We provide the evaluation of **Precision** and **Recall**.

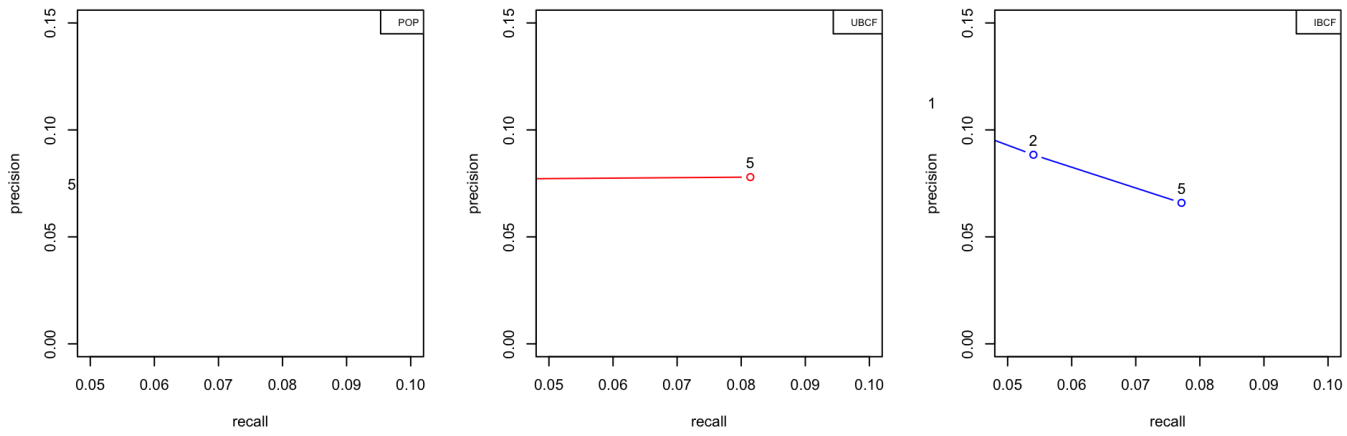

Figure 24 : Recall/Precision for Recommender system methods (Weekend)

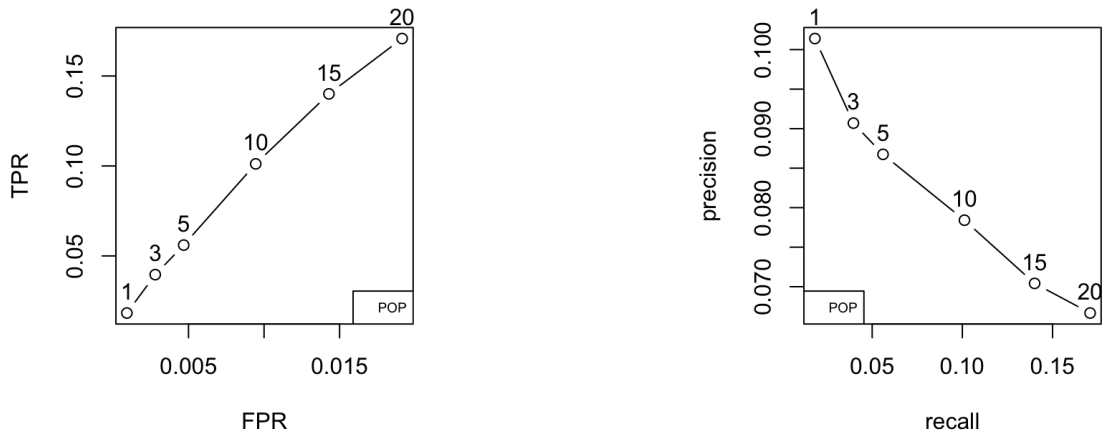

Figure 25 : ROC and precision/recall curves (Weekend Users)

In conclusion, we can state, for what regards binary ratings, that in 3 of 4 different data sets considered that the 2^{nd} and the 3^{rd} user have similar movies recommended for **Association Rules**, the same happens for the 3^{rd} and the 4^{th} user in **IBCF**. The reason for that could be that 2^{nd} and the 3^{rd} saw the same movies while the 3^{rd} and the 4^{th} user saw movies that were very similar (same genre, same ratings, …)

Non-binary context-based approach

The output bellow shows the comparison of the non-binary ratings for the same critic Nell Minow when using the whole dataset for the Top-5 recommendations, the dataset divided in **weekdays** for the Top recommendations and **weekend** Top recommendations, respectively, when using **Popularity**. We can see that recommendations are quite similar for the same user. However, predicted ratings are not.

```
##    m1381    m1067     m769     m674    m1890
## 2.435193 2.377036 2.367250 2.141231 2.030959
```

```
##    m1938    m1392    m1223
## 1.068745 1.741869 1.765340
```

```
##    m1381    m1646    m1067    m1822     m769
## 1.409265 1.587874 1.548523 1.611560 2.001882
```

The output bellow shows the comparison of the non-binary ratings for the same critic Nell Minow when using the whole dataset for recommendations, the dataset divided in **weekdays** recommendations and **weekend** recommendations when using **UBCF** and the Top-5 recommendations. We can see that movie recommendations are very different for the same user depending on the dataset used. This is ways is not possible compare ratings.

```
##     m582     m583     m986    m1992    m1163
## 1.771190 1.717857 1.748968 1.764524 1.717857
```

```
##      m23    m1606    m1381    m1017     m782
## 2.644133 2.493619 2.652640 2.417794 2.615964
```

```
##    m1750    m1418     m118    m1767    m1067
## 1.563836 1.547375 1.439188 1.516931 1.449873
```

Conclusions

Analyzing our recommender systems for the **non-binary** approach we can see that the **precision** measure, which describes what proportion of the recommendations were actually enjoyable for the user, is around or less than 20% what is quite low. In its turn, we also have a lower **recall** (the percentage of movies recommended from all his/hers preferred ones), $i.e.$ less than 10%. This means that less than 10% of the movies the user indeed enjoys were effectively recommended to him. The poor success rate of this approach may be related to the very sparse data set available.

Actually taking into account the non-binary whole set ratings, an empty proportion of 98.7% can be calculated using the equation: $Sparsity = 1 - \frac{\#(ratingsinthedataset)}{(\#critics * \#movies)} = 1 - \frac{51371}{(3489 * 1132)} = 0.987$. We have tried to decrease emptiness proportion by removing from the **non-binary** dataset critics with less than 6 reviews, even though emptiness remained still very high.

Also, a quite poor non-binary rating prediction capacity is observed. Take for instance the **MAE** error that computes the deviation between predicted ratings and actual ratings. We have a **MAE** error in the order of $ca.$ 0.7 - 0.8. This means that a true rating of e.g. 2 can be roughly estimated either as a 2.7 or 1.3: almost either a 3 or a 1, implying in a very big error in the non-binary rating predictions.

Analyzing our recommender system for the **binary approach** we can state that, in general, different methods recommend different movies for the same user, the viceversa (same method recommend different movies for different users) is also true, except for some cases that we discussed in the previous sections.

Observing the graphics we register that the **ROC** curve has always the same behaviour considering different data sets, the **precision/recall** curve for the *Weekend* reviews has a monotonic decreasing behaviour while in the other 3 cases the shape of the curve presents a maximum for Top-5 Popular recommendations, the conclusion one can obtain is that for the *Weekend* is much more complicated to give a good recommendation because the size of the data set is much more reduced than the other cases. An effect is what happens for Ken Hanke; he receives only one recommendation for **AR** and that is because he does much more reviews during the weekdays than in the weekend.

In general is always better to use a **non-binary** set of data because we are able to provide more accurate suggestions for users but in some cases where the data are huge the **binary** approach can be a good way to save memory and money (binary approach is cheaper and faster but less accurate).

References

[1] - Michael Hahsler (2018). recommenderlab: Lab for Developing and Testing Recommender Algorithms. R package version 0.2-3. http://lyle.smu.edu/IDA/recommenderlab/ (http://lyle.smu.edu/IDA/recommenderlab/)

[2] - Asela Gunawardana, Guy Shani. A Survey of Accuracy Evaluation Metrics of Recommendation Tasks. Journal of Machine Learning Research 10 (2009) 2935-2962