Santa Claus - Movie Recommender System

Carlos Soto Garcia Delgado
i6216792

JD O'Hea
i6208128

Sebastien Boxho
i6221844

Ranjani Venkataramani
i6196871

ABSTRACT

This paper explores a combination of Recommender Systems on the Movie Lens 20M dataset. A Collaborative Filtering algorithm, two Content Based algorithms, a Hybrid strategy, and Group Aggregation strategies are implemented. Each of these algorithms have to be evaluated on a subset of the data because of computational and time contraints, thus we had to explore subsetting techniques. Fairness of the Collaborative Filtering and Content Based algorithms are also explored.

INTRODUCTION

The movie recommendation problem has become very relevant since the development of online streaming platforms such as Netflix, Amazon Prime Video, etc. The problem is further popularised with competitions such as the "Netflix Prize" which was an open competition for the best collaborative filtering algorithm to predict user ratings for movies [1].

We aim to tackle aspects of this problem using the MovieLens 20M dataset. We frame this problem as a prediction problem, where given some data about a user we try and predict what rating this user would give to a set of movies. In application then we would recommend the highest predicted rated movie to the user. We also explore the problem of recommending movies to groups.

Successfully creating a Recommender system has many challenges. What exploring a dataset, what features to train a model on, how to subset a large dataset, evaluating a recommender model, analysing if your model contains bias, explaining a model's recommendations, combinations of models, creating groups from your dataset to implement group recommendation algorithms, strategies for group recommendations, and evaluating group recommendations. We explore each of these aspects in the following sections.

DATASET

The dataset chosen for this project is the MovieLens-20M dataset. It is provided by the GroupLens research group and it can be found in the following link: https://grouplens.org/datasets/movielens/

The motivation behind this choice is that the each movie contains a lot of information that allows exploring a content based recommendation algorithm. A second reason is that this dataset is widely used in the reserch community [3]

Features

It contains 20 million ratings of 27,000 movies from 138,000 users. Additionally, 465,000 tag applications are contained, with a total number of 38644 tags. Each movie also comes with its year of release and genres. Lastly, a description of each movie is contained in another dataset that can be easily merged with the MovieLens one.

Data Pre-processing

In order to reduce as much as possible the number of features, lemmatization was applied to our tags using the nltk library. Then tags were regrouped. This reduced the number of tags, however the reduction was not significant. Additonally, stemming and stopword removal were also applied to our tags
A future idea for preproccessing would be to take the word embeddings of the words that form the tag and average them, then build a classifier that would output whether two tags have the same meaning or not (by computing the distance between them). This would allow tags like 'scary' and 'horror movie' to be categorized as the same tag. Due to time constraints, this was not implemented.

Sparsity analysis

A sparsity analysis was conducted. It was observed that a small percentage of movies accumulate the majority of ratings. Additionally the number of tags applied per movie follows a similar distribution (mean=23, variance= 4900). This distribution is also followed by the number of movies that have a specific tag(mean=12, variance 5700).

Data subsetting

Our large dataset (20 million ratings) lead to unreasonable computational time when running some of our algorithms, which pushed us to explore some subsetting techniques for us to be able to carry on with a reduced computational burden. We explored two methods of subsetting, a random subset where we take a random sample of ratings from the original dataset and a recent subset where we take the most recent ratings.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

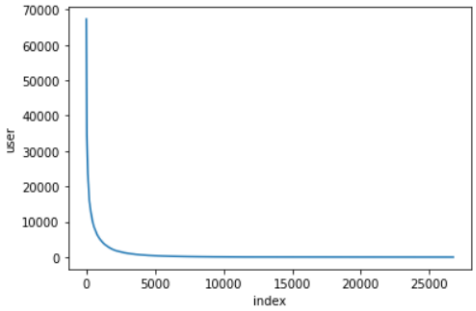

Figure 1. Visualization where on the horizontal axe the movies, and on the vertical axe the number of users who evaluated it

We took a sample of 20% from the original dataset using each method. We then tested the subset to the original dataset using a two-sample Kolmogorov-Smirnov test for to test whether the samples came from the same distribution. We chose the two sample KS test because it is a non-parametric test, i.e. we do not make any assumptions of the underlying data, and also it tests distributions of categorical data. We plotted the frequency of number of movie ratings per movie genre, and tested this distribution. We chose to plot by genres, as opposed to rating number or tags for example, because genre is a major feature of our content based algorithms and it is less sparse than the tags but more sparse than rating value.

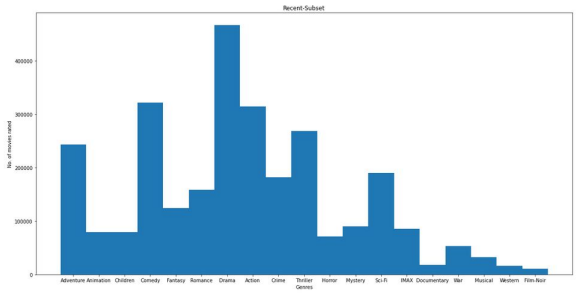

Figure 2. Recent Genres Subset

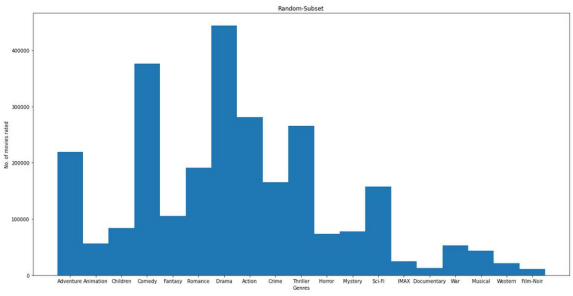

Figure 3. Random Genres Subset

Our null hypothesis is that the two samples, recent and random samples, came from the same distribution. At an alpha of 0.05 we did not have enough evidence to reject this null hypothesis, with a p-value of 0.9.

We chose to use the recent subset for the remainder of our work because of the nature of people, this would be more representative of a user's current preferences compared to a random sample of all of the user's preference.

RECOMMENDATIONS FOR INDIVIDUAL USERS

Three different recommendations algorithms for individuals users are implemented. Content based algorithms and a collaborative filtering algorithm were implemented and a hybrid pipeline algorithm that combines them both.

Content based algorithm

For the content based algorithms the features used are genres, tags and year of release. Plots information was not included on purpose as it would dramatically increase the feature space. A very high number of features must be compensated with much data. Since the recommended system is being run on a single PC with a subset of all the data, this was not possible.

The year feature was normalized to range between 0 and 1. For the tags and genres the information was vectorized where each entry represents a genres or a tag. Additionally, TF-IDF was applied; this allowed to, for example, reduce the importance in the prediction of genres and tags that can be observed in many movies, such as for example 'Drama'. A normal count-vectorizer approach was also run in order to study the effect of the TF-IDF in the recommendations

The algorithm used for the prediction was a K-Nearest Neighbours Regression algorithm where k was set to 5. The distance metric used was the 'Euclidean Distance' as it performs well in high-dimensional spaces.

Collaborative filtering algorithm

A collaborative filtering algorithm was also implemented. The tastes of similar users were used to estimate the ratings for the unseen movies. To implement such algorithm, the UserUser function in the Lenskit library was used. Internally, this libray uses a KNN approach. The minimun number of neighbours is set to 2 and the maximun is set to 15, the library then performs hyperparameter tuning.

Hybrid pipelined algorithm

The pipelined hybridization algorithm is constructed as a cascade of the Content Based algorithm and the Collaborative filtering algorithm. By definition, a cascade hybrid is a sequence of succeeding recommenders, thus the first recommender computes a list of 50 movies which are then given as input to the second recommender. The second one computes a refined list of the same movies and gives a sorted list as output. The list is sorted in descending order from the highest predicted score to the lowest.

Measuring diversity

The content based algorithm considers information on the different movies and on tastes of the user. However, in theory, it fails to provide diverse recommendations. The collaborative algorithm and the hybrid algorithm should be able to provide more diverse recommendations. A study of the level of

diversity of recommendations across different algorithms is performed.
In order to measure the similarity between two movies i and j, the following similarity metric is defined:

$$Similarity(i,j) = \lambda_{tags} * 2 * \frac{|T_i \cap T_j|}{|T_i| + |T_j|} + \lambda_{genres} * 2 * \frac{|G_i \cap G_j|}{|G_i| + |G_j|} \quad (1)$$

where T_i and G_i represent the set of tags for movie i and the set of genres for movie i, respectively. λ_{genres} is set to 2/3 and λ_{tags} is set to 1/3 as a movie has in average more tags than genres.

The diversity of a given list is measured by summing the pair-wise similarity of all movies in the list. Hence, when comparing two recommendations list outputted by two different algorithms, they both must have the same number of movies

RECOMMENDATIONS FOR GROUPS OF USERS

A recommender system for groups of users is implemented. For each member of the group, the ratings for the unseen movies is obtained with the CF algorithm described before. The work of Gartrell 2010 [2] is followed. Gartrell proposes to use different aggregation strategies for different groups in terms of internal strength. Aggregation strategies for 3 different groups are proposed

Groups chosen

It is imagined that 3 different types of groups exist:

1. A couple on a relationship
2. A group of friends
3. A group of people who met recently and do not have a very close relationship

Please note, that in our experiments we do not have such information, but it is believed that if this algorithm was to be implemented and used in production, the system should ask first the groups to identify themselves with one of the above.

Aggregations chosen

The aggregation strategy is based on the internal relationships.

1. For the couple on a relationship, the most pleasure strategy is used
2. For the group of friends, the average strategy is used
3. For group of people who are not very close, the least misery strategy is used.

Least Misery Strategy

The Least Misery strategy looks at the lowest rating for each movie and chooses the one(s) with the highest rating. In some sense, it chooses the best out of the worst, ensuring that nobody is too upset.

Most Pleasure Strategy

The Most Pleasure Strategy looks at the highest rating for each movie and picks the movie(s) with the highest rating. This ensures that only the movies that are truly preferred, "the cream of the crowd", are picked.

Average Strategy

The Average Strategy considers the predicted ratings of all the users and then looks at the mean of the ratings for each movie. The movie(s) with the highest mean is(are) then chosen. This allows for fairness and impartiality.

Fairness

The Fairness Strategy looks at each user in a set order and chooses the movie that has the highest recommendation for each individual, giving everyone a fair opportunity to pick a movie of their choice. This gives everyone a chance to receive recommendations that they would like the most.

Hybrid Strategy 1: Combination of No Misery, Least Misery and Most Pleasure [5]

The First Hybrid Strategy makes use of the No Misery, the Least Misery and the Most Pleasure Strategies. The No Misery Strategy uses a threshold and only movies with their corresponding lowest rating above the threshold are considered for the rest of the method. The minimum and maximum ratings for each movie are then totaled and the movie(s) with the highest combined rating are recommended.

Hybrid Strategy 2: Combination of Fairness and Average [5]

The Second Hybrid Strategy combines the Fairness and Average Strategies. In each round, the highest rated movie for the considered user is picked and in case of ties between movies, the movie with the highest average rating amongst all users is picked.

EXPLANATIONS

The justification and explanation generated by a recommender system has an impact in the user's trust, satisfaction and even effectiveness. As users tend to prefer short and simple explanations [5], we decided to implement a one sentence explanation for every model. For an individual user using the Collaborative Filtering algorithm the explanation is as follows: 'Users with a similar taste liked item x in the past. Check out x'. This helps to convince the user that item x might be to his liking. For the group of users the explanations are based on the aggregation strategy. For the Least Misery and Most Pleasure we used a social-choice based explanation as defined in Barile et al., 2021 [4]. Least Misery: "Item x has been recommended to the group since no group members has a real problem with it." Most Pleasure: "Item x has been recommended to the group since it achieves the highest of all individual group members." For the Average explanation we use a broader formulation "Item x has been recommended to the group since it achieves the highest average rating out of all other items"

EVALUATIONS

Evaluation Methodology

Individual Recommendations

For each of our algorithms recommending to a single user we are using the Mean Absolute Error and Mean Squared Error. We chose these metrics to compare our algorithms because ratings are between 1 and 5, and our predictions are continuous numbers in that range. Our goal is to get as close a prediction as possible to what a user rates a movie so we can recommend to them movies that they will like. We are not taking context into account in any of our analysis. Mean Absolute Error computes the deviation between predicted ratings and the actual ratings, which is the perfect evaluation metric for our goal. Root Mean Squared Error is similar to Absolute Mean Error but places more emphasis on larger deviation, this allows us to get a sense of the deviation of our predictions from the actual ratings.

$$MeanAbsoluteError = \frac{1}{n}\sum_{t=1}^{n}|p_i - r_i| \tag{2}$$

$$RootMeanSquaredError = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\Big(p_i - r_i)^2} \tag{3}$$

We use a simple 5-fold cross validation, where 80% of the interactions are used as training set and the 20% of the ratings used as test set, this evaluation method was chosen to reduce variance and bias in our results. For our Collaborative Filtering algorithm this is run for every user that we have ratings for in the subset. For our Content Based algorithms, we run this on 500 users.

When we split the train and test set there are users in the test set who had no ratings in the train set which the model was created, this is comparable to a cold start scenario. In this case we do not get predictions and we do not include these evaluations in our final metrics as this is not a shortcoming in the model but a shortcoming in our data.

Group Recommendations

For the evaluations of the group recommendations, we use the Mean Absolute Error. We use the average rating among all the users as the standard for comparison. As we're looking at just the absolute errors, any deviation, positive or negative, is considered while calculating the mean error for each aggregation strategy. This allows us to look at the distance between our standard value and the produced result.

Evaluation Results

Table 1 contains the results for our Content-Based and Collaborative-Filtering algorithms. We see the Collaborative Filtering algorithm performs the best in both metrics followed by our Content Based TF-IDF and then CountVec.

	CB (CountVec)	CB (TF-IDF)	CF
MAE	0.794	0.765	0.654
RMSE	0.999	0.951	0.756

Table 1. Table Of Results: Individual User Recommenders

Most Pleasure	Least Misery	Average	Hybrid 1	Hybrid 2
0.25	0.12	0.00	0.06	0.31

Table 2. Table Of Results: Group Recommenders

Looking at the table for the evaluation of the Group Recommendations in table 2, a few surprising but understandable results are found. The Least Misery algorithm outperforms Most Pleasure, one possible explanation is the potentially large variations between individual user preferences as only the top ratings for each movie are considered in the Most Pleasure Strategy. The second hybrid strategy also performs quite poorly, which could, again, be due to the same reason as previously explained: considering only one user's vote for choosing a specific movie. One way to overcome this would be to consider the best and worst ratings, which is exactly what the first hybrid strategy does; also, expectedly, this is the best performing method when we exclude the Average strategy. So we conclude that the first hybrid strategy would be the optimal solution when group satisfaction is the goal.

Measuring diversity

The diversity was measured across the 3 different recommendation algorithms for individual users. The different algorithms were run on 100 random users producing recommendation lists of the same size (10 items). For each of them the average similarity across lists was obtained:

Algorithm	Similarity
Content Based	0.6342
CF User-Based	0.5377
Pipelined Hybrid	0.5442

Table 3. Average similarity observed across multiple recommendation list for different algorithms

As expected, the content based gives more similar recommendations, whereas the collaborative filtering approach and the pipelined hybrid approach both produce lists with a lower level of similarity. The two later ones have a comparable similarity as the pipelined hybrid algorithm uses the collaborative filtering as the first algorithm in the pipeline, this is the one that is in charge of searching most of the movie-space

Fairness Analysis

To test the fairness of our algorithms we decided to investigate the highest errors in our predicted ratings, similar to how we evaluated whether our subsets were fair in the Data subsetting section. We filtered for all movie prediction errors greater than 0.5, and then plotted the the frequencies of movies by genre again. We completed a two sample KS test again with our null hypothesis being that the new sample of high error movies came from the same distribution as the data set we trained our models on. If we do not reject the null hypothesis we can say with a degree of confidence that our algorithm is not bias as our high errors come from the same distribution as our original sample

For our Collaborative Filtering algorithms we got a p-value < 0.01, at alpha = 0.05 we can reject the null hypothesis and conclude that the set of movies that have a prediction error of greater than 0.5 is not representative of the underlying dataset, thus the algorithms are performing better for some subsets of movie genres, and more poorly for others. We see the frequency plot for the Collaborative Filtering movie rating predictions > 0.5 in Figure 4

Similarly, for our Content Based algrorithm we got a p-value < 0.01. At alpha = 0.05 we can again reject the null hypothesis and draw the same conclusions as for the Collaborative Filtering Algorithm. We see the frequency plot for the Collaborative Filtering movie rating predictions > 0.5 in Figure 5

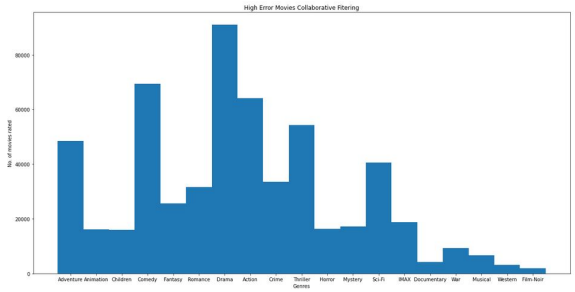

Figure 4. Frequency plot for Collaborative Filtering movie rating predictions error > 0.5

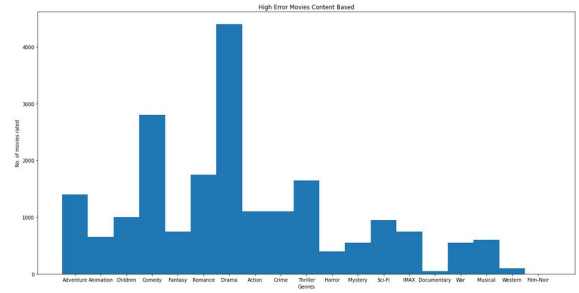

Figure 5. Frequency plot for Content Based movie rating predictions error > 0.5

CONCLUSION

A recommender system was built for a representative subset of the MovieLens 20M dataset. A content based algorithm, as well as a collaborative filtering user-based and a pipeline approach were built as recommendations for individual users. It was observed that the collaborative filtering approach performs better than the content based algorithms for the two error metrics used (MAE, RMSE). A similarity metric was defined to evaluate similarity. The evaluation reflects that the CF approach as well as the hybrid produce more diverse recommendation lists than the Content based ones.

For groups of users different aggregation strategies were proposes depending on how close the users in the group were. Additionally, a hybrid aggregation approach was provided.

Future steps to take in this project could be: evaluating the pipelined hybrid algorithm using hitscore for the outputted recommendation list, generating explanations for the content based algorithm, trying to measure the impact of the dataset size on the accuracy, this is, evaluating our algorithm for different sizes of the data-subset. More work includes a nalysing the sources of bias in our algorithms and performing additional pre-proccesing on tags, such as removing tags that appear less than a certain threshold. We could also evaluate the diversity and fairness of our group recommender strategies, and the diversity of their output.

REFERENCES

[1] Robert M Bell, Yehuda Koren, and Chris Volinsky. 2008. The bellkor 2008 solution to the netflix prize. *Statistics Research Department at AT&T Research* 1, 1 (2008).

[2] Xing X. Lv Q. Beach A. Han R. Mishra S. Seada K. Gartrell, M. 2010. Enhancing group recommendation by incorporating social relationship interactions. In Proceedings of the 16th ACM international conference on Supporting group. *incorporating social relationship interactions. In Proceedings of the 16th ACM international conference on Supporting groupwork* (2010), 97–106. DOI: http://dx.doi.org/10.1145/1880071.1880087

[3] HarperF. Maxwell and A KonstanJoseph. 2015. The MovieLens Datasets.

[4] Shabnam Najafian, Tim Draws, Francesco Barile, Marko Tkalcic, Jie Yang, and Nava Tintarev. 2021. Exploring User Concerns about Disclosing Location and Emotion Information in Group Recommendations. In *HT '21: Proceedings of the 32nd ACM Conference on Hypertext and Social Media*. Association for Computing Machinery, United States, 155–164. DOI: http://dx.doi.org/10.1145/3465336.3475104 The 32nd ACM Conference on Hypertext and Social Media : Hypertext in a Multimodal World, ACMHT21 ; Conference date: 30-08-2021 Through 02-09-2021.

[5] Shabnam Najafian and Nava Tintarev. 2018. Generating Consensus Explanations for Group Recommendations: An Exploratory Study. In *Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization (UMAP '18)*. Association for Computing Machinery, New York, NY, USA, 245–250. DOI:http://dx.doi.org/10.1145/3213586.3225231