Slide 1:

Hello, my name is Taraburca Radu and I will present my version of the project. During the presentation I will explain what I did in the project and how I approached the tasks to complete the project.

Slide 2: Recommender models

Within the project, I had to create the following recommendation models.

1 model that recommends the most popular movies, 2 collaborative filters, a content-based recommendation system and 3 hybrid models.

TopPop model

User-based recommender model

SVD recommender model

Content-based recommender model

Weighted strategy - Hybrid recommender model

Switching strategy - Hybrid recommender model

Meta-level strategy - Hybrid recommender model

Slide 3: Week 1 - Familiarization with the Datasets

The first week of the project was one in which I got used to the data set and in which I analyzed the data set. The data set was already divided previously, so there was no need to divide it into train and test. On these 2 sets of data, I applied some cleaning steps in order to remove missing ratings and duplicate records. Also, to get more information that I will use later, I mapped the initial data sets to the "movies_metadata" set using the "id_map" file that contained the movie id and the metadata id corresponding to the movie.

Besides that, I have made statistics about the users and the movies in the data set, such as the distribution of ratings per movie, the number of ratings per user, the average of the ratings per movie. I also made some histograms to better see the distribution of these statistics.

I calculated the frequency of films with a high rating, greater than 3.

Slide 4: Week 2 - CF Recommender System

This week I dealt with the collaborative filters and the toppop recommendation system.

The toppop recommendation system is a simple recommendation system that only recommends the most popular top k movies.

In making the collaborative filters, I had to choose a specific model for Neighborhood-based model and I chose to do a user-based approach. The decision was quite easy, because I could actually use any approach due to the relatively small number of films and users, but what made me choose this option was the fact that the data set contains some bias towards films popular.

And for the collaborative filter based on a Latent factor model, I chose to use SVD. I chose to use the SVD, because the data set contains sparse and noisy data, and this model can deal with these problems.

To implement these 2 collaborative filters, I used the surprise library.

I also tuned the hyperparameters of the chosen models using 5-fold cross-validation on the training set.

GridSearchCV was used for both models to evaluate the best combination of hyperparameters.

After finding the best hyperparameters, I trained the two models.

Slide 5: Week 3 - Evaluation of Recommender Systems

In the third week, I evaluated the recommender systems using several metrics, including:

RMSE

Hit rate

Precision@k

Mean Average Precision (MAP@k)

Mean Reciprocal Rank (MRR@k)

Coverage

I also had to discusse about the limitations of these metrics and to compared the 3 systems.

The RMSE metric has limitations since it does not take into account the rank of the recommended item and assumes all users give ratings in the same way.

Hit rate, Precision@k, MAP@k, MRR@k, and Coverage are other metrics used to evaluate recommendation systems, each with their advantages and disadvantages.

Comparing three systems (User-based, SVD, and TopPop), TopPop has the best overall performance due to its simplicity and recommendation of popular items. User-based is quick and easy to implement but limited in scalability and accuracy for new users and items. SVD manages data sparsity well but may

suffer from overfitting and requires more data modeling. TopPop is a simple and easy-to-use algorithm that suffers from popularity bias and does not consider user preferences.

Slide 6: Week 4 - Text Representation

In the fourth week, as a first task, I had to do the following preprocessing to clean the overview column from the data set: tokenization, transform to lowercase, remove stopwords, stemming. The nltk library was used for the preprocessing steps.

Initially I applied these steps to the data set that contained only the metadata, but later I also applied these steps to the data set that was made of the movie ratings mapped with the related metadata. After these steps the initial vocabulary of 15754 words was reduced to 9559 words. To represent the movies in vector spaces we used these 2 methods: TF-IDF and word embedding. The cosine similarity was calculated for all the movies for each method. The Pearson correlation coefficient was used to compare the two matrices obtained after the cosine similarity calculation. The correlation coefficient was 0.1906, indicating that the two cosine similarity matrices were not strongly correlated.

Slide 7: Week 5 - Content-Based Recommender System

In the fifth week, I developed a content-based recommender system using the transformed movie overviews.

I represented each user in the same vector space as the items by using an average of the items the user rates.

I calculated the user-item ratings for each movie using a cosine distance.

After that I calculated the following metrics for the system. Precision@5, MAP@5, MRR@5, hit rate, and coverage.

Slide 8: Week 6 - Hybrid Recommender System

In the last week, I had to make 3 hybrid recommendation systems, which use a collaborative filter and the content-based recommendation system. These 3 hybrid systems were based on these 3 strategies:

Weighted strategy

Switching strategy

Meta-level strategy

The recommendation system created for the first strategy (Weighted strategy), is based on the idea of combining the outputs generated by the 2 recommendation systems using some alpha and beta weights. I assigned a slightly higher value to the alpha, which corresponds to the CF model, because it has better metrics than the content-based recommendation system.

The hybrid recommendation system created for the second strategy (Switching strategy), uses the Borda count algorithm to create the final list of recommended movies. Besides that, to solve the tie problem, I used a randomizer to choose a recommendation from the 2 existing lists generated by CF and content-based recommender systems. For a greater degree of randomness, this step of generating lists of recommended movies is repeated a random number of times.

The third hybrid recommender system uses the outputs of the content-based recommender system as input for the collaborative filter.

Besides that, I calculated the evaluation metrics used in the other recommendation systems.

Overall, the first 2 hybrid recommendation models had some good evaluation metrics values. Of all the recommendation systems, the one with the lowest values for the evaluation metrics is the hybrid model created by Meta-level strategy.

That concludes my presentation on our Web Recommender Systems project. Thank you for your attention, and if you have any questions, I am happy to answer them.