EARIN Preliminary Project
19 - Anime Recomender

Jakub Kliszko, Krzysztof Rudnicki

June 11, 2023

The goal of this project is to develop a model for anime recommendation, which takes an anime name as an input and recommends a list of related anime's based on this name.

1 Algorithm description and examples

We will be using the collaborative filtering approach to develop our model. Collaborative filtering is a popular method for making personalized recommendations based on the preferences of other users with similar tastes. In this approach, the recommendation system analyzes a large data-set of user-item ratings to identify patterns and similarities between users and anime's, and uses this information to make recommendations to new users.

We will represent users and anime's data-sets as embedding vectors. Embeddings provide a way to transform high-dimensional data into a lower-dimensional space while preserving relevant relationships.

We have decided to use and test multiple metrics such as

- Cosine similarity
- Mahalanobis Distance
- Euclidean Distance

together with K-nearest neighbors algorithm.

1.1 Embedding users and anime

In order to embed users and anime we must choose which features of the user/anime are the most important for our model
For the user we are restricted to what database offers, database pretty much only offers us what ratings the user gave to an anime, we will compare the performance of our algorithm when:

- Given ratings of the user for all anime that they gave rating to (no matter the watching status)
- Given ratings of the user for all anime that they have *completed*

For anime we will use data:

- User score
- Popularity (this includes popularity itself, number of members and favourites)
- Controversy (Whether the anime gets a lot of 1s and 10s or just 6s)

We have decided to use two different methods to combine data and compare the results
First method is dot product which gives a single number after multiplying two vectors
The second is concatenation which returns a new vector combining first and second vector
As an input for all our metrics (like cosine similarity) we will use vectors of anime and users

1.2 Metrics

Cosine similarity is a measure used to calculate the similarity between two vectors representing items or users. It evaluates the cosine of the angle between the two vectors, indicating their directional similarity. In our use case, cosine similarity can help identify items or users with similar preferences or characteristics by comparing their embedding vectors. Higher cosine similarity values (closer to 1) indicate a stronger similarity between the vectors, suggesting that the items or users are more likely to have similar features or preferences.

Mahalanobis distance is a metric that takes into account the correlations between variables when measuring the distance between two vectors. In our recommendation system, we can leverage Mahalanobis distance to quantify the dissimilarity between the embedding vectors of items or users. By considering the correlations within the embedding dimensions, Mahalanobis distance provides a more accurate measure of dissimilarity. It helps identify items or users that are dissimilar based on their characteristics or preferences, accounting for the relationships between different features.

Euclidean distance is a widely used metric to calculate the straight-line distance between two points in a multi-dimensional space. In our use case, we can apply Euclidean distance to measure the dissimilarity between the embedding vectors of items or users. By comparing the corresponding dimensions of the vectors and calculating the square root of the sum of squared differences, Euclidean distance provides a simple and intuitive measure of dissimilarity. It helps identify items or users that are relatively far apart in terms of their characteristics or preferences, based on the geometric distance in the embedding space.

KNN is a machine learning algorithm that we will use to find the K most similar users to a target user in the similarity matrix. The K most similar users will be our nearest neighbors, and we will use their ratings to generate recommendations for the target user. The number of nearest neighbors (K) that we choose will depend on the performance of our recommendation system on the validation set. We will experiment with different values of K and different weighting schemes for the ratings of the nearest neighbors to optimize the performance of our recommendation system.

1.3 Data used for recommendation

In addition to users and their rating for anime, we will also use measures of:

- Controversy - Anime with lots of '1' and '10 might have the exact same rating as anime with mostly '6' but it is much more controversial (hit or miss) and it can be a good recommendation even though its rating might seem low

- Combined data about popularity - Number of favorites, number of members which are in the group for fans of a given anime and overall popularity on the site
- Ranked - How good the anime is in comparison with other anime's on the site

2 Selection and description of the data-sets

The quality and size of the dataset may greatly affect the performance of the collaborative filtering algorithm. This is why we decided to choose a dataset that was newer and bigger than the one provided in the task. We will be using the Anime Recommendations Database from Kaggle LINK. The dataset contains information about over 17,000 anime's and over 300,000 users.

2.1 Why this data set

There are multiple reasons to use this specific data set over different ones available on Kaggle (including the one proposed in task description)

- It is considered to have 100 % usability by Kaggle
  - It is tagged, has subtitle, description and cover image
  - It has a source, is public and was updated recently (2020)
  - It has the most permissive (CC0) license, good file format, and all files and columns are described
- It is one of the biggest anime databases on kaggle (605 MB)
- It has over 60 code examples of use

2.2 Data-set description

The data-set contains:

- The anime list per user (dropped, complete, plan to watch, currently watching and on hold)

- Ratings given by users to the anime's that they has watched completely.
- Information about anime (exactly 35 columns!), most importantly:
  - Name
  - User score
  - Popularity
  - Members
  - Favourites
  - Number of each particular score (from 1 to 10)
- HTML files containing reviews, synopsis, staff info etc.

There are 5 csv files in total and one HTML folder

2.3 anime.csv

It contains information about all anime scraped
In total it contains **35** columns
Information's that might be in our opinion use-full and are not contained within rest of the files are: Information's that might be use full for our algorithm:

- Number of episodes - Some users might prefer shorter/longer anime's
- When it was aired - Some users might prefer older/newer anime styled
- When it was premiered - Same as above
- Studios - Some users might favourite specific Studios style
- Duration - Some users might prefer anime's with shorter single episode length
- Ranked - Anime that might not be a hit with a certain user but is still considered to be very good can still be a good recommendation
- Popularity - Anime that might not be a hit with a certain user but is very popular can still be a good recommendation

- Members - Same as above
- Favorites - Same as above
- Number of exact amount of scores from 1 to 10 - This can tell how ”controversial” the anime is, anime with lots of ’1’ and ’10 might have the exact same rating as anime with mostly ’6’ but it is much more controversial (hit or miss)

In formations that might be use-full to display:

- Type - Usually whether it is a series (TV or OVA) or movie
- Number of episodes - How much time does it take to watch it
- When it was aired - Start of anime being shown and end of it
- When it was premiered - When the anime started
- Studios - What studio produced it
- Duration - How much time does it take to watch single episode
- Ranked - How good it is in relation to all other anime’s on my anime list

2.4 anime_with_synopsis.csv

CSV file containing anime info for humans
It contains 5 columns:

- (key) mal_id - used to identify the anime
- name - either English or Rōmaji version of the title
- score - average anime score
- genres - list of genres associated with this anime
- synopsis - description of anime

This file will be use-full for displaying the recommendation to user with additional info

2.5 animelist.csv

This file contains information about all users anime lists, no matter their watching status
In total it contains over 105 million rows
There are over 60 million ratings given by users (compared with over 55 for only completed series) It contains 5 columns:

- (key) user_id - random (but persistent through database) user id
- (key) anime_id - used to identify the anime
- rating - what score this user set for this anime (zero is set if user did not set any score)
- watching_status - state ID for this anime in the anime list of the user (see 2.7 watching_status.csv)
- watched_episodes - how many episodes have been watched by the user

This file can be potentially use full especially to determine what users with similar interest are planning to watch

2.6 rating_complete.csv

This file contains information about all ratings given to animes by users who selected watching_status 2 option (complete)
In total it contains over 55 million ratings
It contains 3 columns:

- (key) user_id - random (but persistent through database) user id
- (key) anime_id - used to identify the anime
- rating - what score this user set for this anime (1-10 scale)

This is probably the most important file regarding our project

Figure 1: Entire contents of watching_status.csv file

status	description
1	Currently Watching
2	Completed
3	On Hold
4	Dropped
6	Plan to Watch

2.7 watching_status.csv

Watching status file is used to relate numerical id of watching status to actual textual description of this status
Status number 5 is missing, possibly referring to ”re-watching” status?
Meaning of particular descriptions:

- Currently Watching - Actively keeping up with the series
- Completed - Watched the entire series/film
- On Hold - Stopped watching it but possibly wanting to return to it
- Dropped - Stopped watching it and decided not to return it
- Plan to Watch - Not watched it yet but plan to

For most purposes we will be only interested in status 2 (completed) which tells us that the user checked the anime as already watched
Plan to watch is also interesting since it can be used to recommend anime based on what similar users are interested in

2.8 HTML folder

HTML folder contains HTML scraped info about specific anime's, we will ignore this folder as parsing HTML is a much bigger challenge and outside of the scope of our project

2.9 Summary

There are two most important files, for program inner workings: rating_complete.csv and anime_with_synopsis.csv for showing the recommended anime to user

Figure 2: An exemplary heatmap

Figure 3: An exemplary histogram

3 General plan of tests/experiments

To evaluate and compare the performance of our collaborative filtering model with different parameter configurations, we will use grid search and 5-fold cross-validation.

Grid search will allow us to systematically test different combinations of hyperparameters, such as the number of nearest neighbors in KNN and the regularization strength in the matrix factorization algorithm, to find the best combination that maximizes the model's performance.

Cross-validation will help us to estimate the model's generalization performance and reduce the risk of overfitting to the training data.

To measure the quality of our model's recommendations, we will use precision, recall, and F1-score metrics. Precision measures the proportion of relevant items among the recommended items, recall measures the proportion of relevant items that are recommended, and F1-score is the harmonic mean of precision and recall, providing a balanced measure of the model's performance. We will compute these metrics on the test set and report the average scores across the 5-fold cross-validation.

4 Methods of result visualization

For the visualization, we chose three methods:

- Heatmaps: They are useful for visualizing the similarity between users or animes based on their ratings. They will allow us to identify clusters of similar users or animes and explore how their preferences are related to one another.

- Histograms: They will be used to visualize the distribution of ratings and identifying any biases or patterns in the data. They can help us identify whether the ratings are normally distributed or skewed.

- Precision-recall curves: They show the tradeoff between precision and

Figure 4: An exemplary precision-recall graph

recall at different thresholds, allowing us to identify the optimal threshold for making recommendations. By visualizing the performance of the algorithm in this way, we can identify areas for improvement and generate new hypotheses for increasing the accuracy of the recommendations.

5 Definition of quality measures that will be used

In order to evaluate the quality of our recommendation system, we will use a combination of precision, recall, F1 score, and MAP. Precision measures the proportion of relevant items among the recommended items, while recall measures the proportion of relevant items that were actually recommended. F1 score is the harmonic mean of precision and recall, providing a balanced measure of the system's performance.

MAP is calculated by taking the average of the average precision (AP) for each user. AP is calculated as the mean of the precision values obtained at each relevant item position in the ranked list of recommended items.

To calculate MAP, we need to define a threshold for relevance, which can be either binary (relevant or not relevant) or graded (assigning a relevance score to each anime). In our case, we will use a binary threshold, where an anime is considered relevant if has been rated 8 or higher. We will also calculate MAP for different values of K (number of recommended items), as the quality of the recommendations may vary depending on the number of items displayed to the user.