IRE Project Scope - Team blank

Team Members:

- Ainesh Sannidhi - 2019101067
- Anirudh Kaushik - 2020111015
- Gagan Agarwal - 2021201009
- Tanishq Chaudhary - 2019114007

Project Number 4: Classification/Regression and Recommendation

Project Description:
Given a set of movie metadata, along with plot summary, predict overall rating of the movie.
Given a set of users rating and a list of movies, predict a user specific rating for the movie.

1. Abstract

The rapid growth of data collection has led to a new era of information. Data is being used to create more efficient systems and this is where Recommendation Systems come into play. Recommendation Systems are a type of information filtering system, i.e. they improve the quality of search results by providing results more relevant to the search query and/or related to the search history of the user.

They are used to predict the rating or preference that a user would give to an item based on their previous searches and the other information that the system has related to the user. Most tech companies implement Recommender Systems to a certain extent, such as Amazon for suggesting products to customers, YouTube, Netflix and Spotify to decide which user-specific content to recommend to increase user retention, and any social media platform such as Instagram, Facebook, etc. which use these systems to recommend pages, accounts and ads.

2. Introduction

This project is broadly separated into 2 main tasks:

1. Predict rating of a movie given the movie's metadata and plot summary.
2. Predicting user-specific rating for a movie given that user's ratings for a limited set of movies.

This project is essentially a Recommender System, where Task 1 is a general recommender system where we do not have any information regarding the user who has provided the search query whereas Task 2 will use Recommender Systems that allow us to get query results more specific to the user based on the information we already have on them from their previous movie reviews.

There are three types of recommender systems that we will look at in this project:

- **Demographic Filtering** - They offer generalized recommendations to every user, based on general movie popularity. The System recommends the same movies to users with similar demographic features. The basic idea behind this system is that movies that are more popular and critically acclaimed will have a higher probability of being liked by the average audience. This system will be used as a default to give a rating solely based on the information of the movie.
- **Content-Based Filtering** - They suggest similar items based on a particular item. This system uses item metadata, such as descriptions, actors, director, genre, etc. for movies, to make these recommendations. The general idea behind these recommender systems is that if a person liked a particular item, they will also like an item that is similar to it.
- **Collaborative Filtering**- This system matches persons with similar interests and provides recommendations based on this matching. Collaborative filters do not require item metadata, unlike their content-based counterparts.

3. Literature Review

Matrix Factorization for Collaborative Prediction is a method that uses matrix factorization (multiplying two types of entities to get latent features) in order to determine the relationship between the information of users and the item we are relating this information with (movie ratings in this case). These latent features are in turn used to find the similarity between users and make predictions

Similarity-Based Collaborative Filtering Model for Movie Recommendation Systems is another method that was introduced in order to find similarities between multiple users' reviews for the purpose of Collaborative Filtering. Various similarity metrics such as Pearson correlation, Euclidean distance, cosine similarity and Jaccard distance were tested in this paper in order to determine the closeness between the ratings of users. [2]

The paper, "Attention is all you need." by Vaswani, Ashish, et al. was a ground-breaking paper introducing the transformer model. This model improved upon a lot of the issues of the previous models like LSTM, which suffered from vanishing gradients and long training times. The transformer model parallelized and thus reduced the training time by a lot - allowing a lot more data to be trained. [3]

Soon after, the paper, "Bert: Pre-training of deep bidirectional transformers for language understanding." was released, which was a pre-trained model with bi-directional encodings, allowing for denser meaning representations. BERT has now become the SOTA for any task requiring vector representations. [4]

4. Implementation

4.1. Task 1: Movie Rating Prediction

In order to make predictions based on strictly the user ratings, we will initially use demographic filtering to calculate ratings for movies that we do not have ratings already for without taking into consideration the user-specific information. On these values, we will train regression models (linear regression, stochastic gradient descent, etc.) with respect to the movie-specific values such as runtime, actor, genre, etc. and find a correlation between these values and the ratings. The weights assigned to these parameters will be tweaked after testing as certain values will be much more important than others (such as director possibly being more important than runtime).

4.2. Task 2: Recommendation System

4.2.1. Baseline: Demographic Filtering

Weights will be assigned to movies based on the following rating system (tentative, will be tested and tweaked based on results):

$$\text{Weighted Rating (WR)} = \left(\frac{v}{v+m} . R\right) + \left(\frac{m}{v+m} . C\right)$$

- v is the number of votes for the movie
- m is the minimum votes required to be listed in the chart (results will be skewed with less votes)
- R is the average rating of the movie
- C is the mean vote across the whole report

We already have ‘v’ (vote_count) and ‘R’ (vote_average) and ‘C’ can be calculated as mean(R). ‘m’ will be decided based on the number of movies that we have and how extensive the dataset is (number of votes per movie).

4.2.2. Content Based Filtering

In order to compute pairwise similarity scores between users, we will use all the information presented to us, such as plot, production company, director, actor, runtime and various other factors and tweak their weights and hyperparameters after testing to see which combinations give the most accurate results.

Initially, for the plot description, we will compute TF-IDF vectors telling us the keywords that described each movie in the plot description and compute similarity scores with the Collaborative Filtering based methods we have described above (cosine, euclidean, etc), testing to see which one works the best.

After this, we will also parse the data given to us and extract parameters such as movie runtime, actor and director and use the CountVectorizer() instead of TF-IDF as we will need to increase the importance of these terms as it is more likely that movies with these parameters in common will be more closely related than if the movies’ plots have terms in common.

4.2.3. Collaborative Filtering

Our Content-Based engine suffers from some severe limitations. It is only capable of suggesting movies which are close to a certain movie. That is, it is not capable of capturing tastes and providing recommendations across genres.

These systems recommend products to a user that similar users have liked. For measuring the similarity between two users we can either use Pearson correlation or cosine similarity. This filtering technique can be illustrated with an example. In the following matrixes, each row represents a user, while the columns correspond to different movies except the last one which records the similarity between that user and the target user. Each cell represents the rating that the user gives to that movie. Assume user E is the target.

4.3. Improvements

A major part of our project is to understand the movie plot summary. It is required for both the tasks of movie rating prediction and user-based movie recommendation.

Currently, the models simply use statistical measures such as TF-IDF, however, that only works on the surface level. It assumes there is no syntactic structure to words.

We go a level beyond and instead look at the semantic level of the plot, by using the BERT model. We can consider things like the depth and the content of the story. Thus, using BERT, we can get denser and more meaningful vector representations for the same text.

5. Milestones

30th September: Clean dataset, Explore data (selecting relevant parameters)

7th October: Implement task 1: movie rating prediction using linear regression/SGD/any basic learning method

15th October: Complete task 1 of movie rating prediction

22nd October: Implement task 2 baseline: demographic filtering

31st October: Implement task 2: collaborative filtering

7th November: Implement task 2: content-based filtering

15th November: Complete task 2, fine-tuning model hyperparameters. Improvements if possible.

6. References

[1] Kleeman, Alex, Nick Hendersen, and Sylvie Denuit. "Matrix factorization for collaborative prediction." ICME, 2005.

[2] Raghavendra, C. K., and K. C. Srikantaiah. "Similarity Based Collaborative Filtering Model for Movie Recommendation Systems." *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)* . IEEE, 2021.

[3] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

[4] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).