Recommender Systems

mbkb74
Department of Computer Science
Durham University
Durham, United Kingdom

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

A. Domain of application

This application works on Italian restaurants.

B. Related work review

C. Purpose/aim

The aim of this application is to recommend an Italian restaurant to the user.

II. Methods

A. Data description

The data is from the Yelp dataset [1], this contains a list of the businesses on Yelp, detailing which categories they fall under. It also contains all the reviews, including a written review along with a range of scores. This also now contains a COVID-19 dataset, which lists the changes businesses are making due to COVID-19. There are around 8 million reviews in the review dataset.

B. Data preparation and feature selection

In order to make the data easier to process, along with increasing the accuracy of predictions, it is needed to reduce the number of reviews. First I selected just restaurants, but as this is a large part of Yelp, it only reduced the dataset to 5 million reviews. So I have reduced it to just Italian restaurants, shrinking the dataset to around 470k reviews. In addition, limiting the timescale reduces the impact of tastes changing over time, so I have only included reviews after 2016. This reduced the number of reviews to around 270k, which is enough to give a good sample, but small enough to be practical to process.

C. Hybrid scheme

I have used a meta-level hybrid recommender system where one recommender (content-based filtering) is used to reduce the size of the model of restaurants. This allows for the second recommender (collaborative filtering) to operate on a limited subset of the most likely restaurants. This provides two benefits, one in improving the accuracy of the results by using a second recommender and in improving performance by supplying a short list to collaborative filtering.

D. Recommendation techniques/algorithms

There are two recommenders used here, content-based filtering and collaborative filtering. The content-based filtering works on the text of the reviews to find reviews that use similar words. An example of this is that if someone describes a restaurant as ”cosy”, then it will find other reviews that use the same word. Collaborative filtering looks at the restaurants the user has reviewed and finds people who have reviewed those restaurants in the same way and looks at their reviews of other restaurants to find the best restaurants.

Furthermore the COVID-19 dataset is used to check the list of suggested restaurants and remove those that are closed as the user wouldn’t be able to visit them.

E. Evaluation methods

III. Implementation

A. Input interface

For the input for this application, the user can choose which user they are from a list of 10 randomly selected users from a list of users that have made lots of reviews. This is done so that the recommenders have more data to work with, providing more accurate results.

B. Recommendation algorithm

I find the count of the reviews for each business, and for each user I find how many reviews they have left. Then I only select the 300 most reviewed businesses, this allows for better predictions to be made. I then choose the 100 users who have left the most reviews and take a random sample of 10 to present to the user of the program. Likewise this allows for better predictions. Obviously in a real application you would choose your user from the whole set of users, but this provides more interesting data.

This is where I use content based filtering. From `sklearn` I use the `TfidfVectorisor` to get the TF-IDF matrix, then use the `linear_kernel` function between this matrix and the same matrix, but only including the user reviews. I then use these scores and choose the 300 best matches and merge this with the 300 most reviewed businesses previously found, to generate a smaller list where the restaurant is in both sets.

For each users rating of a restaurant, I group repeated ratings of the same restaurant as the average, this allows for better representation as one person rating the same restaurant 5 stars lots of time shouldn't have more impact than once. I then generate the cosine similarity between all the restaurants as a matrix, using the ratings. For each restaurant the recommender wants to get a prediction for, the recommender loops over the cosine similarities generated for that item. This is to generate an adjusted weighted average, using the formula

$$AdjAvg = ItemAvg + \frac{\sum(cosSim \times (userScore - itemAvg))}{\sum cosSim}$$

This concludes the collaborative filtering section. I then choose the 5 largest scores to present to the user, and remove any which are shown in the COVID dataset as not open.

C. Output interface

The output for this system is a list of 5 restaurants the user would most like, along with their COVID-19 notice so the user knows how to prepare.

IV. Evaluation results

A. Comparison against baseline implementation

B. Comparison against hybrid recommenders in related studies

C. Ethical Issues

V. Conclusion

A. Limitations

B. Further developments

References

[1] Yelp Dataset. https://www.yelp.com/dataset