Creating a Movie Recommendation Engine

Dylan McCardle
COMP 5600
Auburn University
Auburn, AL 36832
dcm0033@auburn.edu

Corey Myers
COMP 5600
Auburn, AL 36832
cam0112@auburn.edu

Evan Sheehan
COMP 5600
Auburn, AL 36832
res0038@auburn.edu

Abstract

The desired outcome of this proposal is to detail the ways in which a program will determine user preferences of different films based on other films' reviews from IMDB. If user provides a short list of films that they like and dislike, the program should output films that the user may also like based on IMDB reviews.

1 Problem formulation

1.1 Input and Output

The input to the program will be five movies that the user likes and five movies that the user dislikes. The output will be five movies that the user may potentially like based on the input movies' reviews.

1.2 Overview

The result will be a movie recommendation engine program. The user will offer one or more movies they enjoy that they would like new recommendations based on. The user will also input a similar number of movies that they dislike. Therefore, the input will be some number of movies that the user enjoys and dislikes. The engine will take these movie titles and find the respective pages on IMDB. The engine will then parse every review for each movie, up to a given boundary, and create a word bank based on these reviews. The word bank will keep track of the occurrences of unique words. The difficulty is determining which words should be counted as unique and which should be ignored. For example, articles such as "the" and "a" will be ignored.

1.3 Identifying data source

The method used to determine the movie recommendations is Naïve Bayes. There are two classes, a like class and a dislike class. Thus, the formula used is $P(c|d) = \frac{P(d|c)*P(c)}{P(d)}$. In this equation, the denominator is ignored since the desired probability is solely in the numerator. Variable c represents the two possibly classes, the like class and the dislike class. Variable d represents the word being tested in the vocabulary of class c. As shown by the example in the link https://medium.com/syncedreview/applying-multinomial-naive-bayes-to-nlp-problems-a-practical-explanation-4f5271768ebf, if the vocabulary is built based on the reviews for the input movies, the program can use the words in the input movies' reviews to compare the probabilities and determine how similar the input movie will be to either class. The formula for each individual word will be the following: (count of the word that appears in like + 1)/((number of words in like) + (number of total distinct words in like + dislike)). The same formula will be used for the dislike class while replacing "like" with "dislike." After obtaining each word's value, multiply all of the words' values together and then multiply by the total probability that the movie is in either class.

2 Database

The program will use IMDB as a data source for reviews and movie-map.com to find similar movies that can be parsed further for review data. The purpose of using movie-map.com for similar movies instead of IMDB is to create an easier and more efficient parse to transverse for results.

3 Scholarly article literature review with reference list

The article "A movie recommendation algorithm based on genre correlations" uses genres and how they combine to recommend movies using an algorithm to try to solve different issues with current movie recommendation systems[1]. They use user input genres and movie ratings to find which movies the user might enjoy. Some problems they encountered were the cold start problem, and the sparsity problem. Since we use a large database of user reviews, the sparsity problem should not be an issue, except in the case of very niche movies with not many reviews. The cold start problem also should not be much of an issue, as movies usually receive the most reviews as soon as they are released, making our usable data very large as soon as the movie is available to be watched.

"A hybrid approach for movie recommendation" proposes creating a movie recommendation engine using a hybrid of content-based and collaborative filtering techniques[2]. Collaborative filtering predicts similarities between the active user and other users. The closest group of similar users is then used to make predictions for the active user. In contrast, content-based filtering is a broad term used to describe the extraction of some features from a source and comparing these features to features of other sources in order to make recommendations. The more similar the features, the more likely a recommendation will be made. This technique is closer to what we will be implementing. However, the article lists the pros and cons of both techniques, leading to the decision to combine them. If we find that our content-based filtering is not providing satisfactory results, this paper may be a useful reference to improve performance. The paper also mentioned a movie recommendation system called MoRe that we may consider using instead of movie-map.com to provide a dataset for recommendations.

References

[1] Sang-Min Choi, Sang-Ki Ko, Yo-Sub Han. "A movie recommendation algorithm based on genre correlations." Expert Systems with Applications, Volume 39, Issue 9, 2012, Pages 8079-8085. ISSN 0957-4174. Accessed 11/5/2019 http://www.sciencedirect.com/science/article/pii/S0957417412001509

[2] Petros Caravelas, George Lekakos. "A hybrid approach for movie recommendation." Multimedia Tools and Applications, Volume 36, Issues 1-2, 2008, Pages 55-70. ISSN 1573-7721. Accessed 11/5/2019 https://doi.org/10.1007/s11042-006-0082-7