Project Report

On

Movie Recommendation System

Under the guidance

of

Ms. Pooja Rana

Lovely Professional University

(2020)

Submitted by: Akshita Gupta Date: April 10, 2020

ACKNOWLEDGEMENT

It gives me immense pleasure to express my deepest sense of gratitude and sincere thanks to my respected guide Ms. Pooja Rana (Assistant Professor), CSE, for their valuable guidance, encouragement and help for completing this work. Their useful suggestions for this whole work and co-operative behaviour are sincerely acknowledged. I also wish to express my indebtedness to my parents as well as my family member whose blessings and support always helped me to face the challenges ahead.

DATE: 10/04/2020 Akshita Gupta

(CSE-KM058-A08)

TABLE OF CONTENTS

ABSTRACT

A recommendation engine refines the data using independent algorithms and propose the most appropriate items to users. It first conquers the past behaviour of a customer and based on that, recommends products which the users are most likely to buy. There are several approaches that are used for our recommended systems. One is content-based filtering, where user's interests are profiled based on the information collected, and then recommend items based on that profile. The other is collaborative filtering, where we try to put similar users together and use information about the group to make recommendations to the user. In this project, a movie recommender system is built based on the MovieLens 1M dataset. We used collaborative filtering method to predict user' s movie rating and we can recommend movies to customers, which they potentially give high ratings according to the prediction. Recommended systems have become ubiquitous in our lives. Yet, currently, they are far from optimal. We try to understand the different types of recommendation systems and analyse their performance by comparing the dataset on the MovieLens. We try to build an extensible model to carry out this analysis. We start by developing and differentiating the various models on a smaller dataset of 100,000 ratings. Then, we try to apply the algorithm so that it is able to handle 1 million ratings. We find that for the smaller dataset, using user-based collaborative filtering results in the lowest Mean Squared Error on our dataset.

INTRODUCTION

A recommendation system is a type of filtering system on the basis of some information which attempts to predict the choices of a user, and make suggestions based on these selections. There is a huge variety of applications for recommendation systems. These have become widely popular over the last few years and are now used in most online platforms that we mostly use.

The products of such platforms vary from movies, music, books and videos, to friends and stories on social media platforms, to products on e-commerce websites, to people on professional and dating websites, to search results returned on Google. Mostly, these systems are able to gain information about a user's choices, and can utilize this information to improve their choices in the future. Due to the advances in recommended systems, users constantly expect good recommendations. They have a low threshold for assistance that are not able to make appropriate suggestions. If a music streaming application is unable to predict and play music that the user likes, then the user will simply stop using that. This has led to a high emphasis by tech companies on improving the commendation systems. However, the problem is more complex than it seems. Every user has unique preferences and likes. In addition, even the taste of an individual user can differentiate depending on a large number of factors, such as mood, season, or type of activity the user is doing. For example, the kind of music one would like to hear while exercising differs greatly from the type of music he'd listen to when cooking dinner. Another problem that recommendation systems have to solve is the exploration vs exploitation problem. They must survey new domains to discover more about the user, while still making the most of what is already known about of the user. There are basic approaches that are used for our recommended systems. One is content-based filtering, where user's interests are profiled based on the information collected, and then recommend items based on that profile. The other is collaborative filtering, where we try to put similar users together and use information about the group to make recommendations to the user. A simple example would be suggesting a movie to a user based on the fact that their friend has also the same choice. There are two types of collaborative models. They are divided into two:

User-based collaborative filtering: In this model products are recommended to a user based on the fact that the products that are liked by user are also the same choices of other users. For example, if John and Jack like the same movies and a new movie comes out that John likes, then we can recommend that movie to Jack because John and Jack seem to have the same taste for the same movies.

Item-based collaborative filtering: These systems selects similar items based on users' previous ratings on some particular products or his choices. For example, if users A, B and C gave high rating to books X and Y then when a user D buys book Y, he also get a recommendation to buy book X as the system identifies book X and Y as similar based on the ratings of users A, B and C.

Content based systems use huge data such as genre, producer, actor, musician to commend items say movies or music. Such a recommendation would be for example recommending Infinity War that featured Vin Diesel because someone watched and rated The Fate of the Furious. Similarly, we can get music recommendations from specific artists because we liked their music. Content based systems are based on the ideology that if you prefer a particular item you are most likely to like the similar item.

<u>TECHNOLOGY USED</u>

1. Python

   Python is an interpreted, simplified, high-level, basic-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design ideology emphasizes code legibility with its remarkable use of significant white space. Its language constructs an object-oriented approach seek to assist programmers write clear, logical code

   for small and large-scale projects. Python is firmly typed and garbage-collected. It provides multiple programming pattern, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its extensive standard library.

2. Jupyter Notebook

   The Jupyter Notebook is an open-source web application that permits you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

APPROACH

There are various types of recommender systems with different approaches and some of them are classified as below:

1. **Content-based Filtering Systems**: In content-based filtering, items are recommended based on comparisons between item profile and user profile. A user profile is content that is found to be appropriate to the user in form of keywords (or features). A user profile might be seen as a set of allocated keywords (terms, features) collected by algorithm from items found suitable (or interesting) by the user. A set of features of an item is the Item profile. For example, consider a scenario in which a person goes to buy his favourite cake ‘X’ to a pastry. Unfortunately, cake ‘X’ has been sold out and as a result of this the shopkeeper recommends the person to buy cake ‘Y’ which is made up of ingredients similar to cake ‘X’. This is an instance of content-based filtering. While our system has done a decent job of finding movies with similar plot descriptions, the quality of recommendations is not that great. "The Dark Knight Rises" returns all Batman movies while it is more likely that the people who liked that movie are more inclined to enjoy other Christopher Nolan movies. This is something that cannot be captured by the present system.

Advantages of content-based filtering are:

- They are useful of recommending unrated items.
- We can easily interpret the working of recommended system by listing the Content features of an item.
- Content-based recommended systems need only the rating of the selected user, and not any other user of the system.

Disadvantages of content-based filtering are:

- It does not work for a new user who has not rated any item yet as much ratings are required content-based recommended evaluates the user preferences and provides accurate recommendations.
- No recommendation of serendipitous items.
- Limited Content Analysis- The recommend does not work if the system fails to

distinguish the items that a user likes from the items that he does not like.

2. **Collaborative-based Filtering Systems**: Our content -based engine abides from some drastic limitations. It is only able of proposing movies which are close to a certain movie. That is, it is not capable of captivating tastes and imparting recommendations across genres. Also, the engine that we built is not really distinctive in that it doesn't capture the personal tastes and tendency of a user. Anyone questioning our engine for recommendations based on a movie will receive the identical recommendations for that movie, regardless of who she/he is.

Advantages of collaborative filtering- based systems:

- Depending on the relation between users which signifies that it is content-independent.
- CF recommended systems can suggest unexpected items by observing similar-minded people’s behaviour.
- They can make real quality assessment of items by considering other people’s experience

Disadvantages of collaborative filtering are:

- Early rated problem: Collaborative filtering systems cannot provide recommendations for new items since there are no user ratings on which to base a prediction.
- Gray sheep: In order for CF based system to work, group with similar characteristics are needed. Even if such groups exist, it will be very difficult to recommend users who do not consistently agree or disagree to these groups.
- Sparsity problem: In most cases, the number of items exceed the number of users by a great margin which makes it difficult to find items that are rated by enough people.

RESULTS AND DISCUSSION

The model represents the performance of various methods on MovieLens 100k (small) data. It represents the performance of user. User Collaborative filtering with the hyperparameter as the number of nearest neighbours (k) on MovieLens 100k (small) data. The best result corresponds to k = 300, as we increase the k to 500 the performance reduces, as we are taking into account almost all the users (500/670) for predicting the rating. When we increase the number of neighbours to 500, even the users with very small similarity values will be included which may introduce noise and thus the deterioration in performance is observed. This increase in capacity may lead to overfitting if we do not use regularization effectively. In the case of 100k dataset, the number of users and items is less and thus even a high value of k does not result in over-fitting. However, in case of the 20 million datasets, since the number of users and items are large, the matrices Q and P are large and the number of tuneable parameters is also large. Hence, we observe that a small value of k, around 100 performs well for the larger dataset. A larger value of k results in poor performance for the bigger dataset. The performance of Content Based and User-User Based Collaborative filtering algorithms are 0.9554 and 0.885 respectively on the 100k dataset. The user based collaborative filtering algorithm gives the best performance.

CONCLUSION

A hybrid approach is taken between content based filtering and collaborative filtering to implement the system. This approach overcomes drawbacks of each individual algorithm and improves the performance of the system. Techniques like Clustering, Similarity and Classification are used to get better recommendations thus increasing precision and accuracy. In future we can work on hybrid recommended using clustering and similarity for better performance. Our approach can be further extended to other domains to recommend songs, video, venue, news, books, tourism and e-commerce sites, etc. In our project, collaborative filtering algorithm is used to predict user's movie rating. The MovieLens dataset, which has 1 million ratings, is selected in our project.

REFERENCES

- https://www.geeksforgeeks.org/python-implementation-of-movie-recommender-system/
- https://medium.com/code-heroku/building-a-movie-recommendation-engine-in-python-using-scikit-learn-c7489d7cb145
- https://www.kaggle.com/ibtesama/getting-started-with-a-movie-recommendation-system
- https://www.datacamp.com/community/tutorials/recommender-systems-python?utm_source=adwords_ppc&utm_campaignid=1455363063&utm_adgroupid=65083631748&utm_device=c&utm_keyword=&utm_matchtype=b&utm_network=g&utm_adpostion=&utm_creative=332602034358&utm_targetid=aud-392016246653:dsa-473406569915&utm_loc_interest_ms=&utm_loc_physical_ms=20471&gclid=EAIaIQobChMIqrmzw43b6AIVBhSPCh1FNgpOEAMYASAAEgLFmPD_BwE
- https://realpython.com/build-recommendation-engine-collaborative-filtering/
- https://data-flair.training/blogs/data-science-r-movie-recommendation/
- https://www.quora.com/How-do-I-create-a-movie-recommendation-system-1
- Koren, Yehuda, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems." Computer 42.8 (2009).
- Bao, Zhou Xiao, and Haiyang Xia. "Movie Rating Estimation and Recommendation." CS229 (2012).
- Ricci, Francesco, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. Springer US, 2011.