NOVATEUR PUBLICATIONS
INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING RESEARCH AND TECHNOLOGY [IJIERT]
ISSN: 2394-3696
VOLUME 6, ISSUE 5, May-2019

AN ADVANCED MOVIE RECOMMENDER ENGINE IMPLEMENTED IN PYTHON

MOLOKWU REGINALD CHUKWUKA
Department of Computer Sciences,
Chukwuemeka Odumegwu Ojukwu University, Uli, Nigeria.
live.reginald@gmail.com

ABSTRACT
These days, we are living during a time of suggestion. Amazon remains on top of things in the online business industry by customizing suggestion of things customers may like based on past requests; Trip Consultant gives diverse inn rankings to various clients; Youtube shows "Related Articles" catch on video page to draw in clients; Netflix accomplishes 2/3 of its film sees by suggestions. Recommender frameworks have turned out to be universal in our lives. However, as of now, they are a long way from ideal. In this project, I explored two approaches of the Colaborative filtering method; the Memory-Based Collaborative filter by computing cosine similarity and the Model-based collaborative filtering using the singular value decomposition (SVD) to understand the different section of collaborative filtering and compare their performance on the popular MovieLens dataset. Which is one of the most common datasets used when implementing and testing recommender engines. It contains over 100 thousand movie ratings ranging from 943 users and a selection of 1682 movies. Execution results are displayed too as well as a discourse on future upgrades.

KEYWORDS: Recommender system, collaborative filtering, content based model, Root Mean Square, Mean Square Error, Singular value decomposition (SVD), Movie Lens.

1. INTRODUCTION

A recommender system (some of the time supplanting "system" with an equivalent word, for example, engine) is a subclass of information filtering system that looks to foresee the "rating" or "inclination" a client would provide for an item.[1][2] They are principally utilized in business applications. Recommender systems are used in an assortment of zones, and are most usually perceived as playlist generators for video and music administrations like Netflix, YouTube and Spotify, item recommenders for administrations, for example, Amazon, or substance recommenders for internet based life stages, for example, Facebook and Twitter.[3] These systems can work utilizing a solitary info, similar to music, or various contributions inside and crosswise over stages like news, books, and hunt questions. There are additionally mainstream recommender systems for explicit fields like eateries and internet dating. Recommender systems have been created to investigate and look into articles and experts,[4] collaborators,[5] money related services,[6] and disaster protection. Because of the advances in recommender systems, clients always anticipate great suggestions. They have a low limit for administrations that are not ready or are not quite precise to make suitable recommendations. In the event that a music gushing application can't foresee and play music that the client likes, at that point the client will just quit utilizing it. This very act has issued a high accentuation by tech organizations on improving their recommender frameworks. Nonetheless, the issue is surprisingly mind boggling. Unlike animals, each human client has various inclinations and preferences. Furthermore, even the flavor of a solitary client can change contingent upon a vast number of elements, for example, state of mind, season, or kind of action the client is doing. For instance, the kind of music one might want to hear while practicing contrasts incredibly from the kind of music he'd tune in to when preparing supper. Another issue that recommender system need to address is exploration vs exploitation problem. They must explore new categories to discover more about the client, while still making the most of what is already known about of the user. Two fundamental methodologies are broadly utilized for recommender frameworks. One is content-based filtering, the other is collaborative filtering. With the end goal of this research, only the two sections of collaborative filtering were assessed.

2. REVIEW OF LITERATURE

Obviously, Collaborative filtering is more spread used than content-based systems because it gives better predictions or results and understanding (from an overall implementation perspective). The algorithm has the capability and potentials to do feature learning on its own, which implies that it can begin to learn for itself what features to use and when to use them. Researchers utilize various measurements to assess proposal quality. Normal measurements fall into two classifications. One is statistical accuracy metric, which tries to evaluate the accuracy of the predicted ratings against the true ratings. Mean absolute error (MAE), root mean square error (RMSE), and correlation between predictions and ratings are representatives for this category. Another sort of metric (decision-support accuracy) includes changing unique numerical rating into paired factors (high/low evaluations) by characterizing a rating edge or threshold. For instance, movies with grades 4-5 are viewed as great quality, while those with 1-3 are low-quality. At that point misclassification rates are processed. Recommender systems are broadly utilized, particularly in the online network, however up to now the application regions of the strategy have been restricted to innocuous basic leadership in the diversion space. For increasingly difficult issues, such as booking an excursion, purchasing protection or financial exchange choices, individuals don't believe the system enough to give it a chance to make these choices for them. One approach to beat this doubt is to give the client a clarification (Herlocker, Konstan, and Riedl, 2000). According to a client, a recommender framework is a discovery, which makes it difficult to comprehend why a certain choice has been made. A clarification can give more understanding into the thinking behind the choice and furthermore gives the client the chance to pass judgment, as indicated by this thinking, regardless of whether to confide in the choice or not. It should make the system progressively straightforward. It has been appeared that clarifications increment the acknowledgment of both master frameworks and recommender frameworks, or all in all, choice emotionally supportive networks (Herlocker et al., 2000; Ye and Johnson, 1995). In the accompanying segments advanced movie recommender framework or system is portrayed.

3. METHODOLOGY

In this section I describe the procedures I took in building my recommendation engine. My research covers the two sections of collaborative filtering which are the Memory-Based collaborative filtering by computing cosine similarity and the model-based collaborative filtering using the singular value decomposition (SVD). This I implemented with Google's Sci-Kit learn tool for python.

3.1 Datasets

I used the famous Movie Lens dataset, which is one of the most common datasets used when implementing and testing recommender engines or systems. It contains over 100k movie ratings ranging from 943 users and a selection of 1682 movies. The ratings are from one to five star. One star signifies that a user does not like the movie and five stars implies that the user loves it.

3.2 Train_ Test_ Split

Recommendation Systems by their very nature are very difficult to evaluate, however the need for split of our data set into two segments, one for training and another for testing is of high importance.

3.3 The Memory-Based Collaborative filtering

Memory-Based Collaborative Filtering approach can be divided into two main branches: **user-item filtering** and **item-item filtering**. A user-item filtering takes a distinct user, find distinguished users that are similar to that user based on similarity of ratings, and recommend items that those similar users liked. In contrast, item-item filtering takes an item, excavates for users who liked that item, and map other items that those users or similar users also liked. It takes things and yields different things as suggestions.

- Item-Item Collaborative Filtering: "Clients or Users who selected this thing likewise checked …"
- User-Item Collaborative Filtering: "Clients or Users who are like you likewise checked.."

In both cases, I created a user-item matrix which built from the entire dataset. Since I had split the data into testing and training I created two [943 x 1682] matrices (all users by all movies). The training matrix houses 75% of the ratings and the testing matrix holds the remaining 25% of the ratings. After I have built the user-item matrix I calculated the similarity and created a similarity matrix. The similarity values between items in Item-Item Collaborative Filtering are measured by observing all the users who have rated both items.

For User-Item Collaborative Filtering the similarity values between users are measured by observing all the items that are rated by both users.

A separation metric normally utilized in recommender frameworks is **cosine similarity.** Where the appraisals are viewed as vectors in n-dimensional space and the likeness is determined dependent on the edge between these vectors. Cosine similarity for users a* and *m can be calculated using the formula below. Where you evaluate dot product of the user vector uk and the user vector ua and divide it by multiplication of the Euclidean lengths of the vectors.

$$s_u^{cos}(u_k, u_a) = \frac{u_k \cdot u_a}{||u_k||\,||u_a||} = \frac{\sum x_{k,m} x_{a,m}}{\sqrt{\sum x_{k,m}^2 \sum x_{a,m}^2}}$$

To calculate similarity between items m* and *b I used the formula:

$$s_u^{cos}(i_m, i_b) = \frac{i_m \cdot i_b}{||i_m||\,||i_b||} = \frac{\sum x_{a,m} x_{a,b}}{\sqrt{\sum x_{a,m}^2 \sum x_{a,b}^2}}$$

I created the user-item matrix. Since I have both testing and training data. the pairwise_distances function was used from google's sklearn tool for python to calculate the cosine similarity. The output ranged from 0 to 1 since the ratings are all positive.

3.3.1 Predictions

Because I created similarity matrices that is, user_similarity and item_similarity therefore I made a prediction by applying the formula for user-based content filtering model:

$$\hat{x}_{k,m} = \bar{x}_k + \frac{\sum\limits_{u_a} sim_u(u_k, u_a)(x_{a,m} - \bar{x_{u_a}})}{\sum\limits_{u_a} |sim_u(u_k, u_a)|}$$

looking at the similarity between users k* and *a as weights that are multiplied by the ratings of a similar user a I normalized it so that the ratings stay between 1 and 5 and, as a final step, sum the average ratings for the user that I trying to predict.

The idea here is that some users may tend always to give high or low ratings to all movies. The relative difference in the ratings that these users give is more important than the absolute values. To give an example: suppose, user k* gives 4 stars to his favorite movies and 3 stars to all other good movies. Suppose now that another user *t rates movies that he/she likes with 5 stars, and the movies he/she fell asleep over with 3 stars. These two users might hold a very similar taste but tender the rating system differently.

When making a prediction for item-based collaborative filtering, there is no need to correct for users average rating since query user itself is used to do predictions.

$$\hat{x}_{k,m} = \frac{\sum\limits_{i_b} sim_i(i_m, i_b)(x_{k,b})}{\sum\limits_{i_b} |sim_i(i_m, i_b)|}$$

3.3.2 Results

There are many evaluation metrics but one of the most popular metric used to evaluate accuracy of predicted ratings is Root Mean Squared Error (RMSE). Applying it on my model

$$RMSE = \sqrt{\frac{1}{N}\sum (x_i - \hat{x}_i)^2}$$

The results were as follows:

Table 1: show the RMSE of the 2 Sections of Memory-based collaborative filtering

SECTION	RMSE
User-based Collaborative Filtering	3.135451660158989
Item-based collaborative filtering	3.4593766647252515

3.4 Model-based Collaborative filtering

Model-based Collaborative Filtering Model-based Collaborative Filtering approach is focused on grid factorization which additional time has gotten more noteworthy introduction and examination, mostly as an unsupervised learning strategy for inert variable deterioration and dimensionality decrease. Matrix factorization is widely in used for recommender systems where it can deal better with scalability and sparsity than Memory-based Collaborative filtering. The goal of matrix factorization is to learn the distinct preferences of users and the distinct attributes of items from known ratings (learn features that identified the properties of ratings) to then predict the unknown ratings through the dot product of the latent features of users and items. When you have a very sparse matrix, with a lot of dimensions, by doing matrix factorization you can restructure the user-item matrix into low-rank structure, and you can represent the matrix by the multiplication of two low-rank matrices, where the rows contain the latent vector. You fit this matrix to approximate your original matrix, as closely as possible, by multiplying the low-rank matrices together, which fills in the entries missing in the original matrix.

I calculated the sparsity level of MovieLens 100K to be 93.7%. A well-known sufficient matrix factorization method is **Singular value decomposition (SVD)**. Collaborative Filtering is formulated by relatively approximating a matrix X by using singular value decomposition method.

$$X = USV^T$$

Given m x n matrix X:

- U is an (m x r) orthogonal matrix
- S is an (r x r) diagonal matrix having a non-negative real numbers on the diagonal
- V^T is an (r x n) based orthogonal matrix

Elements on the diagonal in S are known as singular values of X.

Matrix X can be factorized to U, S and V. The U matrix represents the feature vectors corresponding to the users in the hidden feature space and the Vmatrix represents the feature vectors corresponding to the items in the hidden feature space.

3.4.1 Predictions

I made a prediction by taking dot product of U, S and V^T

3.4.2 Results

Using the MSE (Mean Square Error) method that measures the average of the squares of the errors—meaning the average squared difference between the predicted values and what is actual. With the formula

$$\mathrm{MSE}(\hat{\theta}) = \mathrm{E}_{\hat{\theta}}\left[(\hat{\theta} - \theta)^2\right].$$

I evaluated my predictions

Table 1: show the RMSE of the 2 Sections of Memory-based collaborative filtering

MODEL	MSE
User-based Collaborative Filtering	2.727093975231784

4. CONCLUSION

It difficult to validate my results to other work, because almost all writers use different scope for training and validating their algorithm. Even if the author must have used the same dataset it can still be that the results are stiff to compare because the author may have made different assumptions and their algorithms have different preconditions. In my own case for instance, Memory-based algorithms are easy to implement and produce reasonable prediction quality. The drawback of memory-based Collaborative filtering is that it doesn't scale to real-world scenarios and doesn't address the well-known cold-start problem that is, when new user or new item enters the system. Model-based Collaborative filtering methods are scalable and can deal with higher sparsity level than memory-based models, but also suffer when new users or items that don't have any ratings enter the system.

On the side of the Model-based collaborative filtering, carelessly addressing only the relatively few known entries is highly prone to overfitting. Singular Value Decomposition (SVD) can be very slow and computationally expensive. Probably, in more recent works minimization of the squared error by applying alternating least square or stochastic gradient descent and use of regularization terms to prevent overfitting should be employed.

REFERENCES

1) Cosley, D., Lam, S., Albert, I., Konstan, J., & Riedl, J. (2003). Is seeing believing?:. how recommender framework interface affect users' opinion. SIGCHI conference on Human factors in computing systems, 585–592.
2) Herlocker, J., Konstan, J., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. ACM SIGIR conference on Research and development in information retrieval, 230–237.
3) Herlocker, J., Konstan, J., & Riedl, J. (2000). Elaborating on collaborative filtering recommendations. 2000, ACM conference centered on Computer supported cooperative work, 241–250.
4) Google News Personalization: Scalable Online Collaborative Filtering; Das et al; https://www2007.org/papers/paper570.pdf
5) Intro to Recommender Systems: using Collaborative-Filtering; http://blog.ethanrosenthal.com /2015/11/02 /intro-to-collaborative -filtering/
6) Collaborative Filtering Recommender Systems; Stanford Student project; http://cs229.stanford.edu/proj2014/Rahul %20Makhijani, %20Saleh%20Samaneh, %20Megh%20Mehta, %20 Collaborative%20Filtering%20Recommender%20Systems.pdf