An Analysis of Recommendation Algorithms to Devise a Movie Recommendation Engine and the Wisdom of Crowds

Mahesh Reddy Annapureddy, Karthik Chava, Ushasee Das, Ashish Reddy Podduturi

Abstract

From Netflix to Amazon every online video provider attempts to provide relevant movie recommendations. An efficient movie recommendation system suggests to users a list of movies they may be interested in[1]. This is important to drive customer engagement and interest in the service. Without recommendations users would be inundated with choice, a key problem with online movie libraries. Which approach to movie recommendations however is best? In our paper we attempt to solve this problem. By using movie data provided by GroupLens Research[2], we build out 6 recommendations systems based on different approaches. Lastly, we combine these models to investigate whether, working together, they provide an even better, more optimized approach. The models we investigate in this project include Singular Value Decomposition (SVD), Non-Negative Matrix Factorization (NMF), K-Nearest Neighbors, Co-Clustering, a Neural Network Approach, Alternative Least Squares, and finally an Ensemble method which combines the results from all these models. The idea for this Ensemble method flows from the idea that by seeking the wisdom of crowds, error can be minimized. This type of Ensemble learning method is based upon the intuition that collective knowledge is superior to individual knowledge[3]. By training these models on 1 million ratings data points from the MovieLens dataset, we seek to identify what makes a strong recommendation model and how to attain it based on the algorithms we have learned in the class. Additionally, we evaluate the 6 recommendation systems and our bespoke Ensemble method to ascertain which model is the best. Our goal in this project is recommending to users which movie is worth their time and effort so that they can have the best movie watching experience attainable.

Introduction

With the explosion of data on the internet it has become seemingly impossible to surf the proverbial web without some external aid. Much of this has come in the form of recommendation engines. A recommendation engine is, in its most basic form, an algorithm designed to suggest relevant items to users[1]. From ecommerce to social media, recommendation engines have helped provide a guiding hand to billions of users who, without them, would be lost in the sea of data[4]. In certain industries, like the online video industry, they are very critical. This can be observed by the fact that Netflix offered $1 million dollars to anyone who suggested to it the best collaborative filtering algorithm to predict user ratings[5]. Since that date Netflix has closed the competition to newcomers but it has not stopped us from more deeply investigating this space.

In general, there are two types of recommendation systems, collaborative and content-based methods[1]. Collaborative recommendation engines rely heavily on past user interactions. As filtering engines they seek to understand how past interactions between users and items may influence future behavior. These user-item interactions are stored and are then used to predict how future user interactions may occur. Collaborative filtering algorithms are further divided into a memory-based and a model-based approach[4]. A memory-based approach works directly with interaction values, assuming no model. In contrast, a model-based approach, as the name implies, assumes a "generative" model that explains these interactions and, based upon this, attempts to make new predictions[4]. The strength of these two algorithms is in the fact that no additional information about users is required and, with repeated user interactions, a better model develops[1]. The essential drawback, however, is that because the collaborative model relies heavily on past user-item interaction it cannot predict effectively when dealing with a new user who has no prior interactions[1]. This is termed the "cold start problem."

The other general approach pursued by recommendation systems is content-based[1]. Content based methods use additional information about users or items in order to make recommendations[1]. By using extraneous information, like the users age or gender, or the movies' category or director, the model attempts to build a model based on this information in order to better explain user-item interactions. Through this approach the "cold start problem" is avoided as all that is required is a decent profile of the user[4]. We will not be investigating this approach but will remain focused on comparing various collaborative filtering algorithms.

In our project we will be using algorithms that fall within the collaborative category in order to investigate which approach is superior. Lastly, by combining the algorithms into an Ensemble learning model we will observe whether a combination provides an even better solution.

Data

The movie ratings dataset we are using for our project comes from GroupLens Research[2]. It is a dataset consisting of 1 million movie ratings coming from 6000 users on 4000 movies[2]. This dataset will be used for both the training and testing phase of the project. Approximately 80% of the dataset will be used for training and the remaining 20% for testing. This means 800,000 ratings are used to train the dataset and 200,000 ratings are used to test the dataset. In organizing the dataset, we use the user as the natural delimiter. Thus, 80% of all that user's ratings will stay in the training set whereas 20% of that user's ratings will be kept for the testing set.

Evaluation Metrics

In order to compare the various recommendation engines we will utilize certain metrics. What these metrics are should be explained up front so the reader can understand why we decided one algorithm was superior to another. The goal of our project was to identify a recommendation engine that ably suggested to each user a movie they might desire. Thus, the goal was to avoid suggesting movies that the user had no interest in. This meant that minimizing error was a key aim of these models. The metrics we considered integral to our experiment were Precision, Recall, F-Measure, Mean Absolute Error (MAE), and Root Mean Square Error (RMSE).

Precision attempts to identify what percent of positive identifications were actually right, or said another way, the ratio between the True Positives and all the Positives[7]. It is mathematically defined as: Precision = (True Positive) / (True Positive + False Positive), where True Positive is the number of correctly predicted results and False Positive is the number of incorrectly predicted results[7].

Recall measures how correctly True Positives are identified. It is described mathematically as: Recall = (True Positive) / (True Positive + False Negative), where True Positive is the number of correctly predicted results and False Negative is equal to the number of expected results that were not predicted[7].

There is an inverse relationship or "tug of war" between Precision and Recall suggesting another measure, the F-score. The F-score is the harmonic mean of the precision and recall[8]. It is a measure of the test's statistical accuracy. The F-score is defined as: F-score = 2 * ((precision * recall) / (precision + recall)).

The Mean Absolute Error (MAE) of a model is equal to the mean of the absolute values of each prediction error on all instances of the testing data-set[9]. Thus, it measures the mean magnitude of the errors in a set of predictions.

The Root Mean Square Error (RMSE) is equal to the standard deviation of the prediction errors. It is a measure of how spread out prediction errors are[10]. It is defined as the square root of the average of squared differences between prediction and actual observation.

Investigation

We began our experiment by first dividing the dataset into a training set and testing set. Then we used the training set data to train our 6 algorithms. These training algorithms were then used to predict movies for users in the testing set. We compared these predictions against the known ratings in the testing set and against the top n recommendations for each user. Evaluation metrics including Precision, Recall, the F-score, Mean Absolute Error, and Root Mean Square Error were also computed. Finally, we use these algorithms to construct our Ensemble Learning Model solution and test it as well.

Singular Value Decomposition (SVD)

Singular Value Decomposition is a collaborative recommendation engine technique for decomposing a matrix into three matrices which yield more information concerning the matrix data[11]. It is defined as:

$A=U\Sigma V^T$ where

- U is an m×m orthogonal matrix
- Σ is an diagonal m×n matrix
- V is an n×n orthogonal matrix

Non-Negative Matrix Factorization (NMF)

NMF was popularized by Lee and Seung and became popular because of its ability to extract sparse and easily understood factors from data[12]. As a collaborative recommendation algorithm it is similar to SVD but data and components are assumed to be non-negative. It is defined as:

$X \approx WH$ where X is $n \times p$, W is $n \times r$, H is $r \times p$, $r \leq p$. The product of these two non-negative matrices W and H approximates the non-negative matrix X.[13] This matrix factorization can be used for example for dimensionality reduction, source separation, and topic extraction. One approach to understanding how effective the approximation is comes through the Frobenius norm which is defined as:

$$||\mathbf{X} - \mathbf{WH}||_F^2 = \sum_{i,j} (\mathbf{X} - \mathbf{WH})_{ij}^2$$

K Nearest Neighbors (KNN)

The KNN algorithm, another collaborative filtering algorithm, is based on a simple premise, that similar things are close to each other[14]. It captures this idea of similarity by calculating cosine distances. The smaller the distance the more likely items are to be similar to one another. Thus, by finding the closest training samples to a point, it can predict the label for these based on cosine distances. KNN, though simple, has enjoyed wide success in many classification and regression problems.

Co-Clustering

Co-Clustering is a collaborative recommendation technique that given a matrix A seeks to cluster rows of A and columns of A at the same time[15]. Also called bi-clustering, the goal of co-clustering is to generate bi-clusters or a subset of rows which exhibit similar behavior across a subset of columns. Two map functions take place with co-clustering, rows are mapped to row cluster indexes and columns are mapped to column cluster indexes[15]. Co-clustering is a method of co-grouping two types of entities at the same time based on the similarity of their pairwise interactions[16]. Our co-clustering algorithm is itself executed through optimization algorithms like k-means.

Deep Neural Net

A deep neural network deals with many of the shortcomings of the other collaborative filtering approaches. Matrix factorization creates a number of limitations including the difficulty of using side features beyond the user and movie and the drawback that oftentimes the most popular items will be recommended for everyone regardless of unique tastes[17]. A deep neural network, in contrast, will take into account query features and item features which can help make recommendations more specific for the user. In our model the input layer took in data on movie and user. The following two layers, the embedding layers, were for movies and users, which was then followed by a combination layer to merge into a FlattenedMovies and FlattenedUsers layers. Multiple dropout layers were incorporated to avoid overfitting and multiple fully connected dense layers were added. Lastly, our target variable was the rating matrix.

Alternative Least Squares

Alternative Least Squares (ALS) is a matrix factorization algorithm that runs in parallel fashion and is built for large scale collaborative filtering problems[18]. ALS scales very well and does well with sparse datasets. ALS trains by minimizing two loss functions alternatively. It first holds the user matrix fixed and runs gradient descent with the item matrix; it then holds the item matrix fixed and runs gradient descent with the user matrix[18]. Gradient descent is run in parallel across many partitions of the training data via a cluster of machines.

Ensemble Method

Our suggested Ensemble method is a combination of all the other models we employ. We built this model based upon the idea of the wisdom of crowds. By taking into account many different models, and using their mean results, we can minimize error and hypothetically provide a perfect well-balanced result. It will have all the strengths of our collaborative filtering recommendation engines while minimizing their weaknesses. This model is built by taking the mean of the ratings predicting by each of the individual models. It attempts to prove that combining the results of all these algorithms into one over-inclusive paradigm does indeed minimize error and provide the most useful results for users.

Results

The results are shown in the chart and demonstrate the superiority of the Ensemble learning model. This model shows the best precision and recall, a good F-score, as well as the least error, demonstrated by low Mean Absolute Error (MAE) and low Root Mean Square Error (RMSE). Thus, it is superior to all the other collaborative filtering recommendation engines we examined and it is the one we suggest for creating a movie recommendation engine.

Table 1: Results

Algorithm:	Precision:	Recall:	F-Score:	MAE:	RMSE:
SVD	.683	.683	.683	.684	.872
NMF	.6711	.6711	.6712	.7223	.915
KNN	.6778	.6778	.6778	.705	.894
Co-Clustering	.676	.676	.676	.7186	.9165
Deep Neural Net	0.674	0.674	0.674	.70397	.905
ALS	0.651	0.651	0.651	0.681	0.872
Ensemble Method	.68	.68	.68	.68	.87

Conclusion

Our project does indeed prove the wisdom of crowds. Our Ensemble method outperformed the individual collaborative recommendation algorithms on all metrics. Clearly, by including the results of all the models, error is mitigated due to a balancing of sorts. We engineer the best qualities of a collaborative recommendation engine, while minimizing their drawbacks. Thus, in providing the best movie recommendations to users our project demonstrates an Ensemble solution outperforms all individual models, matching user with movie in the least error prone way.

Bibliography

1) Vijay Kotu, Bala Deshpande, "Recommendation Engines." Data Science (Second Edition). Morgan Kaufmann. 2019. Pages 343-394. https://doi.org/10.1016/B978-0-12-814761-0.00011-3.
2) "MovieLens." GroupLens. 2021. https://grouplens.org/datasets/movielens/.
3) Srinath, Krishnan. "Ensemble Machine Learning: Wisdom of the Crowd." https://towardsdatascience.com/ensemble-machine-learning-wisdom-of-the-crowd-56df1c24e2f5#:~:text=In%20short%2C%20you%20heavily%20research,the%20process%20of%20decision%20making.
4) Rocca Baptiste. "Introduction to Recommender Systems." https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada.
5) "Netflix Prize." Netflix. https://www.netflixprize.com/.
6) "Recommendation Systems." Google. https://developers.google.com/machine-learning/recommendation/overview/candidate-generation.
7) "Classification: Precision and Recall." Google. https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall.
8) "F-Score." Wikipedia. https://en.wikipedia.org/wiki/F-score.
9) JJ. "MAE and RMSE: Which Metric is Better?" https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d.
10) "Mean Square Error." Statistics How To. https://www.statisticshowto.com/probability-and-statistics/regression-analysis/rmse-root-mean-square-error/.
11) Lukoobi, Patrick. "Foundations of Machine Learning: SVD." https://medium.com/the-andela-way/foundations-of-machine-learning-singular-value-decomposition-svd-162ac796c27d.
12) Daniel D. Lee and H. Sebastian Seung. 2001. "Algorithms for Non-negative Matrix Factorization." Advances in Neural Information Processing Systems 13-Proceedings of the 2000 Conference, NIPS 2000. https://papers.nips.cc/paper/2000/file/f9d1152547c0bde01830b7e8bd60024c-Paper.pdf.
13) "Non-negative Matrix Factorization." Stanford. http://statweb.stanford.edu/~tibs/sta306bfiles/nnmf.pdf.
14) Harrison, Onel. "ML Basics with the K Nearest Neighbor Algorithm." https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761.
15) "Co-clustering." ML Wiki. http://mlwiki.org/index.php/Co-Clustering.
16) "Collaborative Filtering: Co-clustering." Data Science Made Simpler. https://datasciencemadesimpler.wordpress.com/tag/co-clustering/.
17) "Deep Neural Networks." Google. https://developers.google.com/machine-learning/recommendation/dnn/softmax.
18) Liao, Kevin. "ALS Matrix Factorization in Collaborative Filtering." https://towardsdatascience.com/prototyping-a-recommender-system-step-by-step-part-2-alternating-least-square-als-matrix-4a76c58714a1.