Movie Recommendation System

Laxman Pottimuthi(lp642)
Ashmitha Shetty (ajs660)
Sreeram Maddinenei (sm2323)
Krishnamurthy Subramanian (ks1437)
Rutgers University

1 INTRODUCTION

The scope of this project is to develop online movie recommendation system using various machine learning techniques learnt during the course, evaluate their performance and accuracy and present the results along with suggestions for further enhancements. We implemented algorithms based on Collaborative Filtering, Sentiment Analysis and content based. We implemented Matrix Factorization based algorithms SVD and NMF. Other collaborative filtering algorithms we used user based KNN and item based KNN techniques. We also implemented another collaborative filtering algorithm through the additional information provided by sentiment analysis of review text in the original data provided and helpfulness of reviews. We have shown results of different experiments based on data split of users(eg: 5core data, 10core data, 20 core data). By Results prove that a hybrid system with the combination of collaborative filtering, content based filtering and sentiment analysis gave better maximisation of precision and recall. Content based filtering is used for better recommendation which also help in recommending to users with cold start.

2 RELATED WORK

Collaborative Based, Content Based and Hybrid approaches are the three famous techniques used for recommending products to users. We propose a hybrid approach which uses sentiment analysis of review text to improve the final predictions. Some research has already been done in this area.[1] has proposed a new hybrid recommendation approach which is built on Spark platform and which proposes to improve the accuracy of recommendations especially in mobile systems.[2] has proposed a new hybrid approach using movie tweets data. They combined content based filtering , collaborative filtering with sentiment analysis of text data. They used movie tweets as their text data.Collaborative and Content Based filtering which we used .[3] proposed an LDA model which uses a hybrid approach. This hybrid approach improved the accuracy of predictions compared to a single recommendation method. It proposed a new LDA approach to calculate customer's preferences on various book topics.

Conference'17, Washington, DC, USA
© 2016 ACM. 978-x-xxxx-xxxx-x/YY/MM...$15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

3 PRELIMINARIES

3.1 Primary data set

The data set that is being considered is Amazon small, 5-core Movies and TV review data (1.7m records) set from Julian McAuley's (UCSD)(data page) site provided as a part of the assignment. 5-core implies each user and movie has at least 5 reviews. This is a json file with following fields in table 1: data set titles

Table 1: Data set titles

Field Name	Type	# unique	Description
reviewerID	string	123,960	Unique reviewer ID
asin	string	50,052	Unique Movie ID
reviewerName	string	123,960	Not used
helpful	list	n/a	Helpful and total votes for the review. Eg. Helpful: [7,9]
reviewText	text	n/a	Movie review provided by the user
overall	int	5	Movie rating 1 ->5
summary	text	n/a	Title of the review
unixReviewTime	long	n/a	Review time stamp (unix)
reviewTime	date	n/a	mm dd, yyyy

3.2 Primary Data Characteristics

(1) Raw format of the data has duplicates and some empty lines sometimes.
(2) It was verified and made sure that there were no duplicates.
(3) Pre cleaning of data is done and Confirmed there were at least 5 reviews for each movie and 5 reviews by each user.
(4) Maximum number of reviews by a reviewer was 2,213 and maximum of reviews for a movie was 2,368.
(5) There were 50,052 unique movies, 123,960 unique reviewers and a total of 1,697,533 reviews.
(6) Primary Data Characteristics
    - Column (movie) spasity - 0.00403
    - Row (user) sparsity - 0.00403

- Overall matrix sparsity – 0.02736

(7) NOTE: Ratings are extremely positively biased. Mean rating: 4.1106. Ratings vs. Frequency histogram is given below. 25th percentile rating is 4.0.

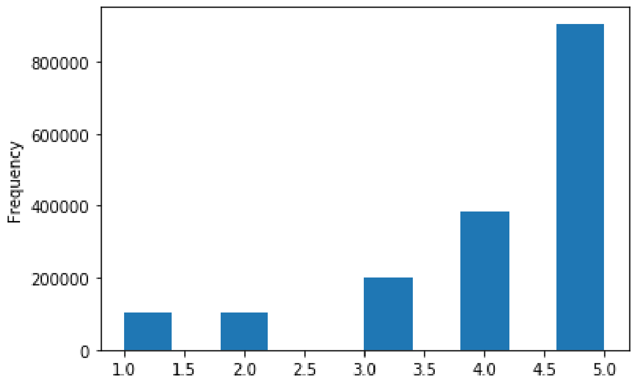

Figure 1: Ratings vs. Frequency histogram

3.3 Supplemental Meta Data

Jianmo Ni's (UCSD) Amazon Movies and TV metadata file that contains 204k metadata records. This is a json file with following fields given in Table 2:

3.4 Supplemental (meta) Data Characteristics

- 21,926 duplicates that were discarded.
- 303 records (20 categories) that were not Movies and TV. Eg. Books, Baby, Grocery etc.
- Several fields had no data. eg. tech1, price, feature, date etc. were mostly NaN.
- Of the 50,052 unique movies in original data set, there are 27,362 with meta data information. 45.33% have no metadata information in the supplemental data set.

4 PROBLEM FORMALIZATION

In this project we decided to create a movie recommendation system. For that we need to build a rating prediction model. Based on the ratings predicted for the movies which user have not rated before, we recommend top n movies. Based on the preliminaries introduced before clean the data to remove the extra overhead while using the calculations. Predict the ratings based on different models and formulate the total predicted score for each user-movie combination based on the framework provided the following sections. The accuracy of this model can be measured using MAE and RMSE. Then we need to create a recommendation list of n movies for every user and evaluate the quality of the recommendations. The quality of our recommendations can be measured using various measures such as precision, recall, F-measure and NDCG.

5 THE PROPOSED MODEL

With the available data, a mini framework was created to embed predictions from three different kinds of prediction algorithms. Collaborative filtering prediction models can be used to predict the ratings of movies which user have not watched yet. We built our model by combining predictions from various collaborative filtering techniques. Different methods can be used to fine tune the ratings predicted. We predicted ratings based on matrix factorization techniques such as SVD and NMF. To add the weightage of bias based on user preferences and items, we also did user based and item based collaborative filtering through KNN classification. Using the movies plot and genres provided in the data, a movie similarity matrix was created. Using the similarity of the user profile and movie-movie similarity matrix, we obtain the predictions for different users. Another technique we implemented is we analyzed the sentiment of user reviews and generate a new rating based on this sentiment. This rating is also used to generate a new prediction using KNN classification. Using weighted combinations of ratings from collaborative filtering, content based and sentiment analysis, a new prediction was generated. Sentiment score helped to give less importance to the ratings that are classified as negative or neutral. After experimenting with different combinations of predicted ratings produced a new rating was generated for the testset. Accuracy and Recommendations are calculated on these new predictions.

5.1 Training and test Data Sets

(1) The entire dataset is divided into training and testing files. For every user 80 percent of his ratings are put in the training data set and 20 percent of his ratings are put in the testing dataset.
(2) During performance testing, different splits between training and test data were explored to measure the model performance as a function of the split ratio.
(3) Predicted ratings are evaluated by calculating the MAE and RMSE.
(4) Based on predicted ratings, a ranking list of 10 movies not yet viewed, is generated for each user.
(5) Quality of recommendation list is validated by comparing the results against rating data in test set using the following measures:
    - Precision: Average for all users of % of testing items in recommendation list for each user. Precision gives the proportion of recommended items that are relevant.
    - Recall: Average for all users of % of items in recommendation list in all testing items for each user. Recall gives the proportion of relevant items.
    - F-measure: F=2*Precision*Recall / (Precision + Recall). The F score can provide a more realistic measure of a test's performance by using both precision and recall.
    - NDCG: Normalized Discounted Cumulative Gain. This is another metric which measures the quality of the ranking of the predictions.

5.2 Rating Prediction

This section describes in detail various methods used to predict and refine ratings for movies not yet rated (or viewed) by the user.

Table 2: Meta data characteristics

Field Name	Type	# unique	Description
category	string	list	type, media, genre etc.
title	string	182,044	Movie / TV Show title
rank	string		Amazon sales rank if available
main_cat	list	21	Main category. 303 records that are not Movies and TV.
asin	string	182,044	21,926 duplicate records
image	blob		4096-dim vector learned by CNN
description	text	n/a	Details of the package contents etc
brand	long	n/a	Creator, director, performer..
also_buy	list	n/a	List of other products bought (ASINs)
also_view	list	n/a	List of other products viewd (ASINs)
price	decimal	n/a	Lot of NaN's
details	html	n/a	Plot etc
feature	text	n/a	Mostly NaN
date	date		Mostly NaN
tech1	text		Mostly NaN

Performance of both the quality of predicted ratings and quality of recommendation list (using the out of band test data) are provided.

5.3 Collaborative Filtering

Collaborative filtering systems recommend movies based on similarity measures between users and/or items. The goal is to find a large subset of those with the highest expected ratings.

- Movies are similar if they were rated highly by many of the same customers.
- Ratings for unviewed movies are predicted based on both similarity measures.
- Correction for movie bias: If the average rating for a movie is much higher / lower than the average for all movies, it is assumed the movie is more / less popular and hence likely to be rated higher / lower by the user who hasn't viewed it. The predicted rating is adjusted up / down as a function of the difference between the average rating for the movie and average rating for all movies.
- Correction for user bias: If the average rating for all movies by a user is higher / lower than the average ratings of all users, it is assumed the user is likely to rate a new movie higher / lower than others. The predicted rating is adjusted up / down as a function of the difference between the average rating of the user and average rating of all users.

The amazon 20 core dataset is split into two subsets where the trainset has 80 pecent of the user ratings and the testset has the remaining 20 percent. In order for the surprise to parse this raw dataset it first had to be converted to a pandas dataframe and a reader object is created with a rating scale parameter set to (0.5, 5.0).

Using the fit() method we train SVD, NMF, knn user based and item based CF algorithms on the trainset and the test() method will return the predictions made from the testset. The rating predictions from each of the method is converted into a dictionary where the key is the reviewerid and the value is a tuple of itemid, actual movie rating and the estimated rating from each of the algorithm. All the prediction ratings for the movies are averaged and merged into a single dictionary.

5.4 Sentiment Analysis of Reviews

Sentiment analysis is the process of analyzing, processing, summarizing, and reasoning the emotional text. Movie reviews contain users preferences and feelings about the movie and these emotions also affect the choice of other users who see the reviews. Users analyze their personal experience, choose useful reviews (rated as helpful), remove misleading or harmful reviews and ultimately make their own decisions. These decisions are influenced by sentiment of the majority of reviews they find useful for a movie.

Supervised or Unsupervised learning methods can be used to determine emotional polarity of the review into positive, negative or neutral.

Preprocessing: We first prune the dataset by filtering out the unwanted columns. We then clean the data by removing all the empty reviews. Since we are using sentiment as our target we cannot have any null values, so these are dropped. Using the actual movie ratings we classify the movies into 3 classes: positive, negative or neutral.

In order to capture the analysed sentiment of the review, Valence Aware Dictionary and Sentiment Reasoner (VADER) is used sentiment analysis. There are other alternatives to VADER like texblob which can be used. As VADER is highly efficient for reviews and social media text, VADER is used to calculate the sentiment score of the each review with respective to each user.

- Stop words are removed from reviews.
- Lexical features were combined using five general rules that embody grammatical and syntactical conventions for expressing and emphasizing emotional intensity and results were lemmatized.

- VADER process the lemmatized review and returns a compound score between -1 (very negative) and +1 (very positive). This score combines the positive, negative and neutral scores.

Ratings are predicted and recommendation lists generated for each user using:

- Normalized VADER compound scores (expected to perform poorly).
- Linear combination of User Ratings and VADER compound scores.
- Non-Linear combination of User Ratings and VADER compound scores

Helpfulness information is used to correct ratings prediction by sentiment analysis.

- If there are no feedback for the review, sentiment rating is NOT changed.
- If there is feedback for the review, then sentiment rating is scaled by ratio of helpful feedback to total feedback received.

5.5 Content Based Filtering

- Description and category from Supplemental metadata are combined into a single text plot.
- Resulting text is cleaned by removing punctuation, articles and other unnecessary data.
- For each movie, an important bag of words is extracted.
- Movie / Word matrix is vectorized using nltk.countvector package.
- NOTE: Countvector is used instead of TF-iDF since TF-iDF performs poorly when used on movie plots which have frequently occuring information like genre.
- Similarity matrix is calculated using cosine similarity from the vectorized Movie / Important Word matrix.
- This will give us information about the word /importance relevance to a movie.
- Weighted average scores are calculated for movies watched and rated by the user and this information is used to calculate individual scores for movies user hasn't watched.
- To calculate Cold Start movie score, movies are given average ratings.
  $ColdStartMoviescore = averagemovierating * similarityscore$.

5.6 Hybrid prediction

Different similarity scores are calculated from different models as explained above. For getting collaborative score, we average out the different values from all the collaborative prediction models. For content based, weighted average of the movies similarity and movie rating given by the user is calculated for each movie. It can be done by dot product of the movie-movie similarity and user-movie rating matrices. Once new crosstab matrix is achieved, divide the each column by sum of similarities to get weighted average. These scores are added linearly with the sentiment score produced for for the test set to maximise F-measure. We have done experiments for the same, to get the linear combination of scores.

5.7 Tools/packages used

For the implementation part we have used surprise package which is a python scikit library for building and analyzing the performance of our hybrid recommender system. To perform sentiment analysis of the reviews we have used Natural Language Toolkit (NLTK) a NLP library in python.

6 EXPERIMENTS

To find the best train and testset for our algorithm we experimented with different split sizes and calculated the mae and rmse. Following is the acuuracy graph that we obtained for different split sizes.

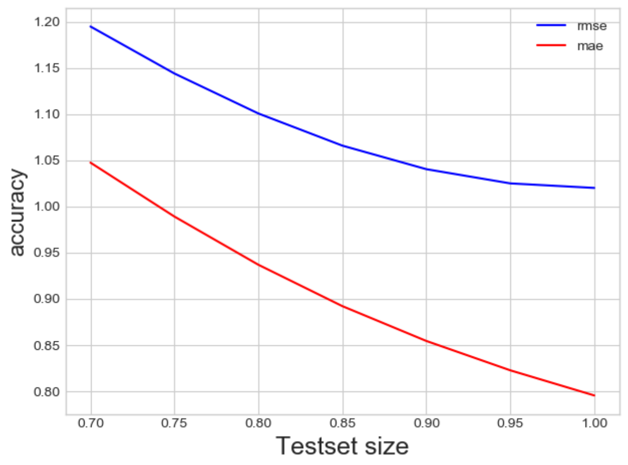

Figure 2: Accuracy vs. Testset size

As we can see rmse and mae is the lowest when we use the entire dataset for training. But this results in overfitting problem and poor generalization of the testdata. So from experiments we figured that a 80-20 train test split would be a good fit for the model.

To evaluate the quality of the recommender system we calculate the precision and recall for the top 5 recommended movies for each user. Precision and recall can then be averaged over all users into a single value for comparison purposes. Figure 3 is the graph of precision vs recall for some of the prediction algorithms. The precision for our recommender system evaulates upto 85%. Here we can interpret that 85% of our recommendations are actually relevant to the user. With a recall of 76% we can interpret that 76% of the relevant items were recommended in the top-5 items which is better than any collaborative filtering algorithm.

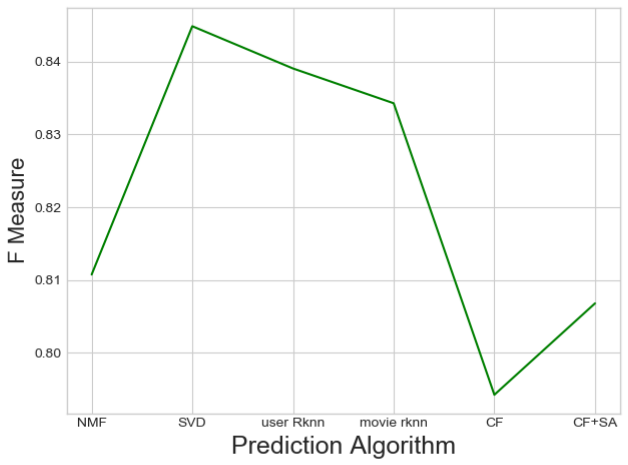

Figure 4: F Measure vs. Prediction algorithms

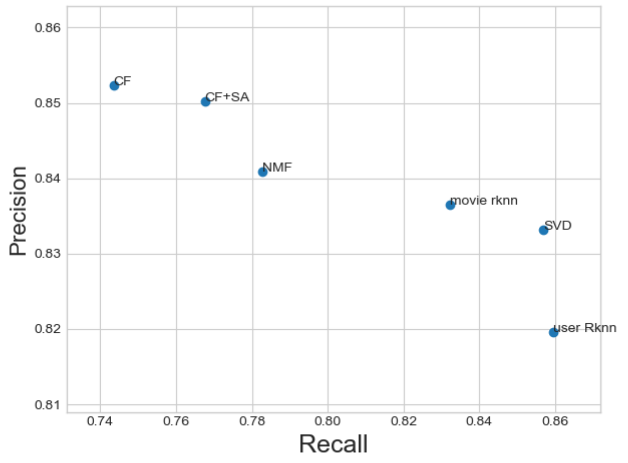

Figure 3: Precision vs. Recall

The table shows all the evaluation metrics computed for our recommendation system for different train-testset split sizes.From the above results for F measure we can conclude that integrating the predictions from different collaborative techniques along the sentiment score gives a better performance. This is also represented in Figure 4 which plots the f measure values for all the prediction algorithms.

To further increase the performance of our recommender system we have also tried to combine content based predictions. Though this runs perfectly on a smaller dataset it fails to execute on the 5 core dataset due to the limited RAM size available. The results that we collected from the smaller dataset shows that by aggregating collaborative filtering and sentiment score with the user's movie preferences gives us more precise recommendations.

'

7 CONCLUSIONS AND FUTURE WORK

The combination of various techniques led to better accuracy in our rating predictions. Sentiment score added to collaborative score and content based score gave the the better fine tuning on ratings predictions. While recommending the movies not watched by anyone, content based helps in recommending such movies. In the future we can take the ratings from social media and get the better sentiment score. We also had memory issues while calculating scores for content based since calculating cosine similarity matrix was very huge. We can use autoencoders in deep learning to reduce the feature reduction and calculate the similarity scores between the movies. Some relevant work was done HFT and VBPR which can give better predictions. Those can be incorporated in the ratings to give better predictions.

REFERENCES

(1) Wang Y, Wang M, Xu W. A Sentiment-Enhanced Hybrid Recommender System for Movie Recommendation: A Big Data Analytics Framework. Wireless Communications and Mobile Computing. 2018;2018:9. doi:10.1155/2018/8263704
(2) Kumar S, Halder S, De K, Roy P. Movie Recommendation System using Sentiment Analysis from Microblogging Data. arXiv.org. November 2018.
(3) Lingfang Zheng, Suling Jia, Qiang Wang. A Hybrid Recommendation Method Based on Feature for Offline Book Personalization
(4) Richter L. Jure Leskovec, Anand Rajaraman, and Jeffrey D. Ullman . Mining of Massive Datasets . Cambridge, Cambridge University Press . Biometrics. 2018;74(4):1520-1521. doi:10.1111/biom.12982

Table 3: Performance measures of our hybrid recommender system for different testset size

	RMSE	MAE	Precision	Recall	F Measure	NDCG
CF	0.9549	0.7052	0.85229426	0.7435	0.7942	0.9823497247462206
CF+SA	1.0202	0.7955	0.850189110556903	0.7675811793973146	0.8067760475567759	0.9776410183003299
CF+0.95SA	1.0251	0.8227	0.8557398171238213	0.7557137443405854	0.8026223737931213	0.9530289003522902
CF+0.9SA	1.0405	0.8545	0.8641645885286471	0.7387655598706295	0.7965600206625028	0.9284280542946934
CF+0.85SA	1.0660	0.8922	0.8732564422277351	0.7146850419370061	0.7860532938511504	0.903839151600943
CF+0.8SA	1.1008	0.9370	0.8855098365197869	0.6814422387749138	0.7701879527384576	0.8792622069674269
CF+0.75SA	1.1441	0.9892	0.9014588528678094	0.6349585045761232	0.7450956764848181	0.8546972482711284
CF+0.7SA	1.1950	1.0476	0.92035	0.56930	0.7034623	0.830143977025318