Heinz 95-845: Movie Recommender System Applied Analytics: the Machine Learning Pipeline

Agnes Huang angesh@andrew.cmu.edu
Heinz College of Information Systems and Public Policy
Carnegie Mellon University
Pittsburgh, PA, United States

Wendy Chiang pingjouc@andrew.cmu.edu
Heinz College of Information Systems and Public Policy
Carnegie Mellon University
Pittsburgh, PA, United States

Wei Wang ww5@andrew.cmu.edu
Heinz College of Information Systems and Public Policy
Carnegie Mellon University
Pittsburgh, PA, United States

Abstract

Recommendation systems nowadays has become an important feature in lots of consuming company. When customer surfing the website, the item that being recommended to them correctly will help to increase companys profit. It provides information based on users activity, items characteristics, etc. Netflix has built a good mechanism utilize Singular Value Decomposition (SVD), this project will choose SVD as the baseline and compared it with SVD plus plus and KNN. The dataset that was used contains 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users and we will choose the best algorithm by compare the metric conducted on datasets.

1. Introduction

MovieLens is a web-based recommender system and virtual community that recommends movies for its users to watch, based on their film preference using collaborative filtering of members movie rating and movie reviews. Therefore, this website contains the market data including users, movies and ratings from a wide range period of time. In addition, Netflix is a company that handles big collection of television programs and movie, it is very important for them to keep users since users have freedom to cancel subscription anytime. In order to do that, the recommendation system is being researched and provide users with programs and movies that can fit their best taste. By using recommendation system, the sales of different product can be increased. It is important that the recommendation system can provide user with accurate product, therefore, a system with advanced algorithm works better.

The recommendation system will consider not only the information about users, but also

the items that user consumed. The previous researches have analyzed several algorithms for recommendation system, including Popularity, Collaborative Filtering, Content-based Filtering and Hybrid Approaches. The collaborative filtering method focus more on user activities and their patterns to conduct the recommendation. This project will focus on a supervised learning process for Movie Recommendation System. The likely outcome will be a recommender system giving recommendation based on users current movie preference, including the movie they have watched and the features of these movies. We compare the KNN with SVD and SVD-plus-plus, using the data set from Movie Lens. These methods require minimal knowledge engineering eorts to produce good enough results, by analyzing similarity among users, its recommendation system is built based on users network. In our study, we will put more emphasis on user him/herself, since for some website, there are lots of user dont friend with other user but still need to get proper recommendation.

The target population is movie audience from 1995 - 2016. The outcome is top 10 movies recommended for user. Firstly our recommend system will calculate the rating of particular movie for certain user, and choose the top 10 scores movie. We used measure RMSE (Root Mean Square Error) for this project. Since we decide to use item-based error by comparing the dierence between average recommending list per user and the corresponding actual list, we can easily get RMSE for this type of predictive analysis.

SVD is going to be our baseline model. Since, during AAML lectures we were exposed with other machine learning approaches like Latent Dirichlet Allocation (LDA) and ensembling and boosting techniques like Gradient Boosting, we would like to explore the opportunity to apply them on real-world dataset and study more about diverse methods through implementation. In terms of the size of this analysis, we did not form a hypothesis to specify the Type I and Type II errors in our study, but luckily we found movie lens data has 20 million ratings and 465,000 tag applied to 27,000 movies, which was updated till 2016 by 138,000 users. We believe this data size should be sucient for our analysis. By comparing SVD, SVD-pp and KNN, we can see what algorithm works best in the recommendation system for MovieLens datasets.

2. Background

The Collaborative Filtering Algorithm focus on the similarity of two user, and it assumes that they will behave similar in the future (Breese and Kadie (1998)). This method has a good indication in the very social-environment application. In our case, movie recommendation, works well. It considers users activity in regardless users demographic information and the movie characteristic information. For example, Mike and I gave high rate in several movies, for another movie k, I gave movie k a good rating, then it is more possible that Mike will also give a high sore to that movie.

In the paper of Leidy (Leidy Esperanza MOLINA FERNNDEZ), the researcher explained the method as follow. In our dataset, we have a collection of user ui, and collection of movie mj, where i is from 1 to n, and j is from 1 to m. for each user ui to movie mj, there will be a rating happen, so the rating value rij will be in the (i,j) position in the matrix of n*m.

	m1	m2	...	mj	...	mm
u1	r11	r12	...	r1j	...	r1m
u2	r21	r22		r2j		r2m
u3	r31	r32		r3j		r3m
...	...			...		...
ui	ri1	ri2		rij		rim
...	...	...		...		...
un	rn1	rn2		rnj	...	rnm

Figure 1: matrix of users and movies

Collaborative method also contains several types: memory-based techniques, model-based techniques and hybrid methods that combine the precious two mentioned. This project mainly utilizes the model-based techniques, which using machine learning algorithms to predict users rating of unrated items. Among which the most popular option is Singular Value Decomposition (SVD), the general equation can be expressed as =U*S*Vt, and this is also the baseline of our model, which will be compared with SVD pp and KNN method. The detail of SVD will be elaborated in section 4 Method.

3. Data Set

3.1 Description

The data set we are using obtained from MovieLens website (GroupLensResearch). The MovieLens, which collected by GroupLens Research, contains the rating data sets from MovieLens website and over various peroids of time. There are different dataset size options, we choose the latest small dataset that containing 100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users. The latest update is on 9/2018.
In the data set, users were selected at random for inclusion. All selected users had rated at least 20 movies. No demographic information is included. Each user is represented by an id. Only movies with at least one rating or tag are included in the dataset. These movie ids are consistent with those used on the Movie Lens website. The dataset including four documents: ratings.csv, tags.csv, movies.csv and links.csv, we only utilize ratings.csv and the rating parameters in our research this time.
In rating file, each line after the header row represents one rating of one movie by one user, and has the following format: userId, movieId, rating, timestamp. Ratings are made on a 5-star scale, with hald-star increments (0.5 stars 5.0 stars). Since we only analyze ratings at this time, only one dataset has been used, while for further research, movie.csv, tag.csv can all be included to analyze whether the specific features can contribute to the recommendation model better.

3.2 Pre-processing

In model building process, we need to have training and testing dataset. For our recommendation system, we assume that the user is someone already have records in the rating system. Therefore, it is important to keep train and test data consistent on number of users. In another word, when split datasets, it is split based on user instead of the whole data. In the preprocessing, we kept every user id, and for one specific user, split the dataset into 80

percent as train data, 20 percent as test data. After splitting, double confirming that the number of users in both datasets are the same.

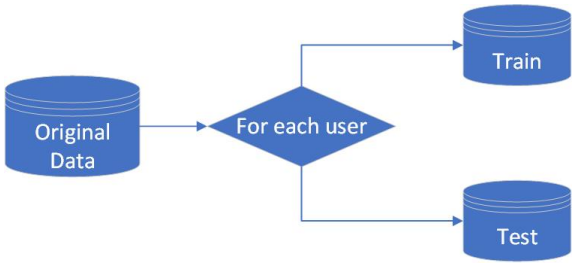

Figure 2: train test split process

The figures 2 below show the idea, training data and testing data will contains same user id information, but the test data will have different movies and their corresponding ratings. These can be utilized in the evaluation of prediction for test dataset.

userId	movieId	rating
1	1	4
1	3	4
1	6	4
1	47	5
1	50	5
1	70	3

Figure 3: Example of Train Data for user 1

userId	movieId	rating
1	101	5
1	110	4
1	151	5
1	157	5
1	163	5
1	216	5

Figure 4: Example of Test Data for user 2

4. Method

There are three methods we used: SVD, KNN and SVD plus plus.

4.1 Model 1: SVD

SVD maps both users and items to a joint latent factor space of dimensionality f, such that user-item interactions are modeled as inner products in that space. In plain language, it assumes a user's rating of a movie is composed of a sum of preferences about the various aspects of that movie. Accordingly, each item i is associated with a vector q_i, and each user u is associated with a vector p_u. For a given item i, the elements of q_i measure the extent to which the item possesses those factors, positive or negative. For a given user u, the elements of p_u measure the extent of interest the user has in items that are high on the corresponding factors, again, positive or negative. The resulting dot product captures the interaction between user u and item $i$the users overall interest in the items characteristics.

However, much of the observed variation in rating values is due to effects associated with either users or items, known as biases or intercepts, independent of any interactions. Thus,

it would be unwise to explain the full rating value by an interaction of the form $q_i^T p_u$.

For example, a pitfall of collaborative filtering data is the large systematic tendency for some users to give higher ratings than others, and for some items to receive higher ratings than others. After all, some products are widely perceived as better (or worse) than others.

Instead, the baseline the system tries to identify is the portion of these values that individual user or item biases can explain, subjecting only the true interaction portion of the data to factor modeling. This approximate user us rating of item i, which is denoted by r_{ui}, leading to the estimate:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u$$

Figure 5: SVD Formula of Estimated Rating

Therefore, our goal is to minimize the following regularized squared error:

$$\sum_{r_{ui} \in R_{train}} \left(r_{ui} - \hat{r}_{ui}\right)^2 + \lambda \left(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2\right)$$

Figure 6: SVD squared error function

4.2 Model 2: KNN

K Nearest Neighbor method is derived from a basic nearest neighbor approach and extended from basic model. In order to determine the rating of user u for movie i, we find other users that similar to user u, and based on the users ratings on this movie, we get the ratings on movie i. The basic collaborative filtering algorithm of KNN is set as the following figure:

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k(u)} \text{sim}(u, v) \cdot r_{vi}}{\sum_{v \in N_i^k(u)} \text{sim}(u, v)}$$

Figure 7: KNN Formula of Estimated Rating

- uid The (raw) user id. See this note.
- iid The (raw) item id. See this note.
- r-ui (float) The true rating rui.

- est (float) The estimated rating rûi.
- details (dict) Stores additional details about the prediction that might be useful for later analysis.
- k (int) The (max) number of neighbors to take into account for aggregation
- min-k (int) The minimum number of neighbors to take into account for aggregation. If there are not enough neighbors, the neighbor aggregation is set to zero (so the prediction ends up being equivalent to the mean u or i). Default is 1.

As the algorithm name indicate, KNN finds the k nearest neighbors of each movie under the similarity function and use the weighted mean to predict this rating. The v¡-N(I, k(u)) shows that v belongs to k most similar users to movie i, and these users have rated movie i. There might be options that the KNN is item-based, but in our case, since we only utilize information about users rating on movie instead of the features of movie itself, therefore, user-based method is more appropriate.

4.3 Model 3: SVDpp

SVDpp is an extension algorithm of SVD, which captures implicit feedback and provides an additional implication of user preference. Look at the formula below, the difference is the addition $|I_u|^{-1/2} \sum_{j \in I_u} y_j$ factor. This factor includes the implicit effect, which is opposed to SVD that has only explicit effect p_u. To interpret this, we could think that the probability that a user "likes" a movie that he/she rated are higher than a not-rated movie. This is an "implicit" information, and SVDpp captures it while SVD does not.

The overall equation of the estimated rating of user u for item i for SVDpp is in figure 8. And like SVD, the parameters are learned via SGD on the regularized squared error objective (Koren (2008)).

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T \left(p_u + |I_u|^{-\frac{1}{2}} \sum_{j \in I_u} y_j \right)$$

Figure 8: SVDpp Formula of Estimated Rating

5. Experimental Setup

In order to make the recommender system more accurate, we processed the data with the following steps for the data split process and feature selection. In addition, we will explain how we evaluate the outcome of the predicted result in this section.

5.1 Population Description

We have four columns in our raw data: UserId, MovieId, Rating and Timestamp. On figure 1, the dataset has total 100,838 rows and the average rating is around 3.5 point. Looking

at the figure 2, it captures the count for each unique UserId. On figure 2, it shows that the dataset has total 610 unique UserId, and each UserId has seen 165 movies. (Varies from 20 to 2698 for each UserId.)

	UserId	MovieId	Rating
Count	100,836	100,836	100,836
Mean	326.12	19435.29	3.50
Std	182.61	35530.98	1.04
Min	1	1	0.5
25%	177	1199	3.0
50%	325	2991	4.5
75%	477	8122	4.0
Max	610	193609	5.0

Figure 9: Dataset Description

	UserId	Count
Count	610	610
Mean	305.5	165.3
Std	176.23	269.48
Min	1	20
25%	153	35
50%	305	70
75%	457	168
Max	610	2698

Figure 10: Dataset Unique ID Description

5.2 Feature Choice

Since every pair of (userId, movieId) is unique and we only need user ID, movie ID and rating for our analysis purpose, we drop timestamp column. Therefore, after processing, we have UserId, movieId, and rating in our prcessed data.

5.3 Split data

To make the model more accurate, when we split the dataset into training and test data, we make sure that every user exist in our training and test dataset. We believe that this split could receive better predict result.

5.4 Evaluation Criteria

When evaluating the result, we calculate the average rating of each UserId for both test data and predicted result. Then we calculated Root Mean Square Error (RMSE) to compare the predicted averate rating and an the known rating.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

Figure 11: RMSE Formula

The overall process flow can be summarize in Figure 12 below. We first remove timestamp column and then split data into train and test while making sure each UserId occurs in

both dataset. Secondly, we train the data with three models: SVD, KNN, SVDpp. Finally, we evaluate the models by calculating RMSE and also compare their performances.

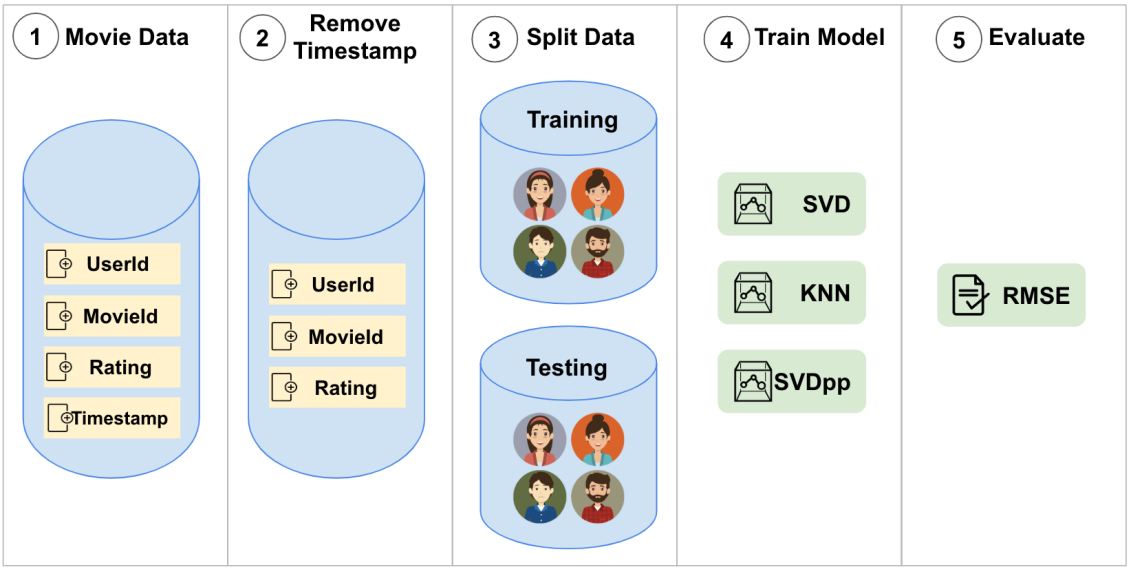

Figure 12: Experiment Flow

6. Discussion and Related Work

	Algorithm	test_rmse	fit_time	test_time	avg_ratings
0	SVDpp	0.884233	179.707006	5.333150	3.653219
1	SVD	0.893880	3.484540	0.199427	3.660058
2	KNNBaseline	0.962982	0.408815	1.280196	3.642679
3	average	NaN	NaN	NaN	3.661196

Figure 13: matrix of users and movies

Our metrics summary showed that SVDpp had better RMSE score on test data, which was slightly smaller than SVD original and much smaller than KNN with baseline. KNN with baseline showed its limitation to cope with the systematic tendency of extreme ratings compared with other matrix factorization-based algorithms. Despite its good performance, SVDpp was also a time-consuming model to train, which took 50 times more than SVD needed to fit all our training data. In our experiment, we concluded that SVDpp is the winner among three; however, one should be mindful of the large training time on the same scale of data it needs to get better performance than the original SVD.

7. Conclusion

In this experiment, especially the comparison of SVD and SVDpp, we found the more information a recommender system model could include, such as implicit preference, the better the RMSE score we could get. For further research on improving the winner model, other information sources that could contribute to the formula may be user attributes, for example, demographics. Again, for simplicity consider Boolean attributes where user u corresponds to the set of attributes A(u), which can describe gender, age group, Zip code, income level, and so on (Yehuda Koren and Volinsky (2009)). A distinct factor vector y_a in the dimentionality f corresponds to each attribute to describe a user through the set of user-associated attributes:

$$\sum_{a \in A(u)} y_a$$

Figure 14: User attributes

After adding user attributes into the formula, we get:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T \left(p_u + |I_u|^{-\frac{1}{2}} \sum_{j \in I_u} y_j + \sum_{a \in A(u)} y_a \right)$$

Figure 15: Extended formula

This setting ideally will enhance the user representation in our collaborative filtering method.

References

David Heckerman Breese, John S. and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence. UAI98. Madison, Wisconsin: Morgan Kaufmann Publishers Inc., pp. 4352. ISBN: 1-55860-555-X.*, 1998. doi: http://dl.acm.org/citation.cfm?id=2074094.2074100.

GroupLensResearch. Movielens dataset. `https://grouplens.org/datasets/movielens/latest/`. Accessed on 2019-4-26.

Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. 2008.

L. Leidy Esperanza MOLINA FERNNDEZ. URL `https://beta.vu.nl/nl/Images/werkstuk-fernandez_tcm235-874624.pdf`.

Robert Bell Yehuda Koren and Chris Volinsky. Matrix factorization techniques for recommender systems. 2009.