Amazon Recommendation System

Jason Cai, Matthew Eng and Amol Srivastava
Department of Computer Science
Rutgers University

ABSTRACT

Recommender systems have become an important part of our digital lives, influencing our choices in various domains. This paper dives into the design, implementation, and evaluation of an item-to-item recommender system on Amazon data. Our findings highlight the importance of understanding the underlying data characteristics and selecting appropriate recommendation techniques to achieve optimal results.

KEYWORDS

SVD, collaborative filtering, item-to-item, MAE, RMSE, precision, recall, f-measure, NDCG, training, testing, dataset, utility matrix, similarity matrix, sparse matrix, prediction, recommendation

1. INTRODUCTION

For this project, we were tasked with creating a recommendation system based on one of the two provided datasets. We used concepts that we had learned from class (SVD, Collaborative Filtering) and further research to create models that made accurate predictions and recommendations. To be specific, we had to complete the following tasks: Data selection and preprocessing, Rating Prediction, and Item Recommendation. For each task, we measured how well our code ran through the following metrics: MAE, RMSE, Precision, Recall, F-measure, and NDCG.

2. RELATED WORK

When preparing to solve the problem at hand, we researched ideas and concepts that we could apply to our resolution. From the start, we decided that we wanted to implement item-to-item collaborative filtering.[1] So we reviewed previous lectures and searched the web for ways we could implement item-to-item recommendations to our project. In a more general sense, for inspiration, we also researched Causal Collaborative Filtering.[4] During this phase, we also briefly researched Neural Collaborative Reasoning to give us more context in terms of learning and reasoning.[3] Finally, after testing multiple models, we ultimately decided to go with SVD. So we furthered our understanding of SVD through past lectures and a helpful article that we discovered on the web.[2]

3. PROBLEM FORMALIZATION

$$\sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}}$$

Figure 3.1

Figure 3.1 shows the formula for the Root Mean Square of Errors (RMSE). In this formula, N denotes the total number of data points, each $\hat{y}_i$ denotes the predicted value for the i^{th} data point and each y_i denotes the actual value for the i^{th} data point.

$$\frac{|y_i - \hat{y}_i|}{N}$$

Figure 3.2

Figure 3.2 depicts the Mean Absolute value of Errors (MAE). This is another method for us to measure the accuracy of our recommendation algorithm.

$$\frac{\text{\# of True Positives}}{\text{\# of True Positives + \# of False Positives}}$$

Figure 3.3

Figure 3.3 describes the method for calculating precision. The best precision is 1 i.e. when there are no false positives.

$$\frac{\text{\# of True Positives}}{\text{\# of True Positives + \# of False Negatives}}$$

Figure 3.4

Figure 3.4 describes the method for calculating recall. For both precision and recall higher values are better.

$$\frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Figure 3.5

Figure 3.5 is the formula for the traditional f-measure. Notice that it is dependent on both precision and recall that were defined previously.

$$\frac{\sum_{i=1}^{k \text{ (actual order)}} \frac{\text{Gains}}{\log_2(i+1)}}{\sum_{i=1}^{k \text{ (ideal order)}} \frac{\text{Gains}}{\log_2(i+1)}}$$

Figure 3.6

Figure 3.6 shows the formula for normalized discounted cumulative gain (NDCG). Here "Gains" represents the relevance score of each item and k is the number of items that are being considered.

4. THE PROPOSED MODEL

To efficiently and effectively solve this problem, we sequentially solved the different tasks. To complete the first task, we implemented the Pandas library to be able to read the JSON file and converted it into a Dataframe. The Dataframe was split and used to implement an item-item similarity model to aid us in predicting the product ratings in the testing sample. However, we found that this item-item similarity model wasn't the best fit in generating recommendations for each of the users, so we decided it would be best to implement a collaborative filtering model through matrix factorization and singular value decomposition. We felt that this was the most efficient model for generating recommendations for users because it allowed us to account for users in addition to items rather than strictly relying on items.

5. EXPERIMENTS

At the end, our results from the experiments varied from task to task. Starting off with task 1 (Data selection and preprocessing), we successfully split the Industrial and Scientific dataset into training and testing datasets. All of our datasets had 12 columns. In terms of rows, the entire dataset had 77071 rows. The training and testing datasets had 62307 and 14764 rows or roughly a 80.8% and 19.2% split, respectively. Our second task was making rating predictions, so we took the training dataset we created and implemented an item-item similarity approach. We found this approach most suitable in predicting ratings as it allowed us to find items similar to the one we are predicting and take the weighted average corresponding to their similarity. This model did fairly well in predicting ratings as the calculated MAE (Mean Absolute Error) and the RMSE (Root Mean-square Deviation) were 0.5796 and 0.9146 respectively. With some experimentation of the item-item similarity model we found that it was not the ideal approach in making user recommendations. So with some experimenting we came to implement a collaborative filtering model through matrix factorization and singular value decomposition. Through the use of the NumPy arrays we were able to map users to items, perform matrix operations and implement our own SVD operation. Using our implementation of SVD we were able to filter and find the top recommendations for each of the users. Our resulting precision, recall, f-measure and NDCG are 0.01, 0.074, 0.017 and 0.067, respectively. We found that these low metrics were due to the lack of individual user ratings in the testing dataset. Many users in the overall dataset had only rated a few times. So when the dataset was split into testing and training, many users would only have one or two ratings in the testing dataset. This lack of individual rating from some users made it very difficult to create an accurate top 10 recommendations for them as the sample size for them was simply too small.

As a result of this our metrics suffered when our recommendations were compared to the testing dataset.

6. CONCLUSIONS AND FUTURE WORK

Overall, this project was extremely beneficial to us as it gave us a better understanding of important data science and machine learning topics. In terms of the task at hand, it was engaging to figure out which dataset to choose from and which algorithms/models to use to find the best results in an efficient manner. Incorporating useful Python libraries into our code also enhanced our coding and resource-finding abilities. This project also helped us build a great framework/template for recommendation systems that can be applied to any dataset. Given the dataset that we used mainly had ratings as the key attribute, the resources/libraries that we were able to implement, and the overall quality of data (e.g. lack of individual user ratings in the testing dataset), we felt that our results/metrics were satisfactory. However, for future cases/work, choosing datasets that had more helpful attributes for its items could introduce Content-Based Filtering as a valid algorithm to make accurate predictions and recommendations.

ACKNOWLEDGEMENT

We would like to thank the professor for his in-depth and detailed lectures that helped us understand the concepts of data science and that helped us complete this project. We would also like to thank the TAs for their helpful recitations and discussions that further expanded our understanding of complicated concepts.

REFERENCES

[1] Muffaddal Qutbuddin. Comprehensive guide on item-based recommendation systems. Retrieved from https://towardsdatascience.com/comprehensive-guide-on-item-based-recommendation-systems-d67e40e2b75d.

[2] Jake Tae. Recommendation Algorithm with SVD. Retrieved from https://jaketae.github.io/study/svd/.

[3] Hanxiong Chen, Shaoyun Shi, Yunqi Li, and Yongfeng Zhang. Neural Collaborative Reasoning. Retrieved from https://arxiv.org/abs/2005.08129.

[4] Shuyuan Xu, Yingqiang Ge, Yunqi Li, Zuohui Fu, Xu Chen, and Yongfeng Zhang. Causal Collaborative Filtering. Retrieved from https://arxiv.org/abs/2102.01868.