Recommendation Systems

Phil Nowak

May 15, 2014

1 Abstract

Recommendation systems are used to predict what a person will want. These systems are helpful in showing users only items they would want, which in turn increases sales for companies. There are a variety of types of recommendation systems, each excel in different areas. Recommendation systems are continuing to be improved upon over time.

Contents

List of Figures

2 Introduction

The vast size of the internet gives people to access a large volume of information. E-commerce sites like Amazon have millions of items too choose from, music sites like Spotify, contain millions of songs, most people are connected to some sort of social media. It can be hard to find what your looking for with these vast sizes. To solve this companies started created ways to filtering out unwanted things and suggest goods that you may want. These are called Recommendation Systems. Over time, these systems have become increasing accurate in making these predictions. There are a variety of types of recommendation systems as well as different ways to implement them.

3 Start of Recommendation Systems

Before the widespread use of computers and the internet, people relied on brick and mortar stores to purchase the items they needed. Brick and mortar stores have a limited amount of space and, therefore, are able only to carry a limited amount of goods. The decision of what goods to carry would simply be based on what was selling at the time or by store owner's preference. This meant there was a small selection from which customers could choose. However, this changed when widespread use of computers and the internet came around. Sites like Amazon and ebay started, and they could store a nearly infinite amount of items on their virtual shelves. This created some problems, however. With so many options, online customers had a difficult time finding the items they would want; because online stores want to sell as many items as they can, they needed to find a way to filter out the user's unwanted items. Online stores had the additional goal of showing items to customers that they would want to purchase which, if done successfully, would drive up store sales [10]. These factors inspired the demand to create recommendation systems.

4 Demographic Systems

Many of the early recommendation systems were basic and did not give personal recommendation to the customer. Oftentimes these recommendation systems were versions

of demographic filtering, which makes recommendations based on personal attributes of the user [4]. These attributes can include age, race, gender, careers, region, and many other personal traits. Recommendations are then made from these characteristics. Demographic recommendations fail to take into account actual user interest, however. An example of one such system was the Grundy, which took personal information from the user and then found books that the user may like based on predetermined stereotypes of the user's attributes [4]. This system gave a general idea for the user; however, more accurate ways to make recommendations were known. Use of more accurate recommendations would raise buyer confidence in the recommendations and therefore increase sales.

5 Content and Collaborative filtering

Content filtering and collaborative filtering are examples of recommendation systems which are able to give more accurate recommendations. Both of these systems take into account specific user interests and are able to give personal recommendations with this data. Collaborative filtering works by finding other users with similar interests and then making recommendations based on these interests [8]. Content based systems make recommendations based on items that are similar to the user's item likes [1]. While both these systems can make personalized recommendations, there are some drawbacks to both. Collaborative filtering requires a vast amount of user data on interests so that accurate similarities can found between users. It also requires a large number of users so there most likely will be people with similar interests to compare. This problem is known as cold start [9]. The problem that content based systems face is that they can only make recommendations on similar types of items [1] While this personalization was an improvement on making accurate predictions, there were still improvements that could be made.

6 Hybrid Systems

The combination of collaborative and content systems, known as hybrid systems, was found to be a way to improve on making accurate predictions. Hybrid systems are able to give recommendations by utilizing the large variety of the collaborative system and avoidance of the cold start issue by using elements of the content system [2]. One problem caused by these systems is the scalability problem. The scalability problem is that as the amount of users and user data increase, more computational power is required to make recommendations [9]. This can be seen as both a positive and a negative. More user data means better recommendations but at the cost of performance. There are still areas that can be improved upon though use of hybrid systems. Many of these improvements require a large amount of computing power, though, for the number of computations that are required. As computer technology improves, so do the number of calculations that can be

done to make recommendations.

7 Further improvements

Computers are becoming faster; according to Moore's law the speed of computers doubles almost every two years. Computer parallelism is also becoming more popular, and data parallelism scales well with the large amounts of data utilized in many recommendation systems. Faster data processing has a few benefits as well. One benefit is that it can more quickly make recommendations to the user. The longer it takes to make recommendations, the less likely the user will wait around to inspect them. Companies such as Amazon have so much user data that it is impossible to compare the current user's interests to every other person's data in a timely matter, thereby allowing only a fraction of the comparisons [10]. Increasing the processing power will allow for a greater percentage of user data to be compared, which will create more accurate recommendations. Another advantage to faster computations is that they allow for more complex algorithms in recommendation systems. One such example is the addition of dimensional reduction. Dimensional reduction finds common characteristics among items in the user's likes and makes a single rating for these items [10]. Examples of such groups could be books by specific authors, or art by certain artists. In this way, recommendation systems can rate people as similar even if their interests are not the exact same.

8 Technical Analysis

There are a variety of approaches to building recommendation systems, one of which is called Collaborative Filtering. This system is the most prevalent recommendation system used today [13]. Collaborative Filtering works by comparing a user's interests with the interests of other users, and then by finding users who share the most similar interests. The likes of these similar users are then compared in order to find things of interest for the original user.

9 K-Nearest Neighbors

One collaborative filtering method used for generating recommendations to the user is the K-Nearest Neighbors method. This method involves finding other users with tastes similar to those of the active user. After locating these other users, recommendations are then calculated for the active user based on the likes of the other users. [7]

10 Find Nearest Neighbors

10.1 Pearson Correlation

The nearest neighbors to the active user can be found by a variety of algorithms. An example of such an algorithm to find the nearest neighbors is called the Pearson Correlation Coefficient (PCC). This algorithm takes each item ranked by both users and

finds the distances between those items [9]. These distances determine how similar the interests of the two users are. The formula outputs values between 1 and -1. A value of 1 means that the two users are in perfect agreement; the larger the value, the more similar are the tastes of the two users [9]. The value 0 means that there is no correlation between the two users. Negative values mean opposite tastes between the users, and -1 means that the two users are perfect opposites. So the active users nearest neighbors are those with the highest values. This is the PCC algorithm:

$$R_{x,y} = \frac{\sum_{i=1}^{n}(X_i - X_m)(Y_i - Y_m)}{\sqrt{\sum_{i=1}^{n}(X_i - X_m)^2}\sqrt{\sum_{i=1}^{n}(Y_i - X_m)^2}}$$

$R_{x,y}$ is the rating between the active user and a user, which is a value between 1 and -1. XM and YM are the mean ratings of user X and user Y. The mean is calculated to compensate for the different ways that users vote. Ideally, the user average would be the middle number in the ranking scale; however this is often not the case. [9]. This means that if a user tends to rank everything high, those high rankings will not be as meaningful as the high rankings of a user who averages low rankings. It is for this reason that the PCC is good for systems where there is discrepancy in how the users might rank.

There can be a slight problem with this method, however. Running this formula requires two passes through the ranking list of the users. The first pass is to calculate the mean, and the second one is to run through the actual formula. If the number of items being ranked is small or if obtaining a timely recommendation is not important, then this formula is fine. However, speed of results matters today in many recommendation systems. Many users are not going to wait around to see what a computer has to tell them. However, there is another version of the PCC that only requires one pass through the data. This second formula makes an approximation of the mean rankings of each user. The end results wind up being very similar [3]. This is the formula:

10.2 Shortened Pearson Correlation

$$R_{x,y} = \frac{\sum_{i=1}^{n}(X_i Y_i) - \frac{\sum_{i=1}^{n}(X_i)\sum_{i=1}^{n}(Y_i)}{n}}{\sqrt{\sum_{i=1}^{n}(X^2) - \frac{\sum_{i=1}^{n}(X)^2}{n}}\sqrt{\sum_{i=1}^{n}(Y^2) - \frac{\sum_{i=1}^{n}(Y)^2}{n}}}$$

$R_{x,y}$ is again the rating between the active user and a user with a value between 1 and -1 [3]. N is the number of

ratings that both users have in common.

10.3 Comparison

To show the comparisons between the two formulas, take a look at the following graph. The X axis shows the number of nearest neighbors that both formulas will find. The Y axis shows the percentage of nearest neighbors that were found with both versions of the PCC. The data of the users and their rankings were taken from MovieLense. There are approximately 6,000 users and 4,000 movies to be ranked. Both versions of the PCC were run on 100 randomly selected users. Then, the nearest neighbor lists of each user were compared to find the percentage of same neighbors of the same neighbors on each list. The percentage on the graph indicates the average percentage of each of the 100 users.

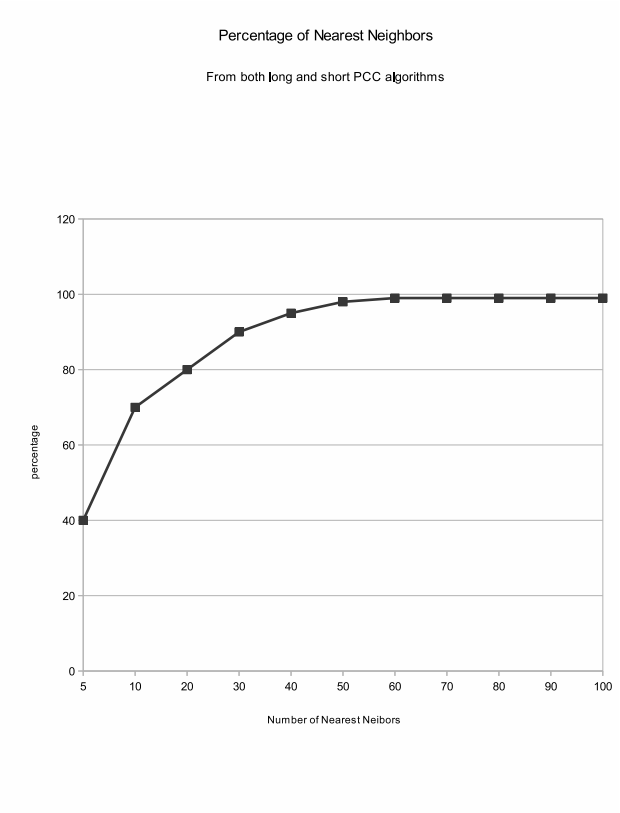

Figure 1: Comparison of long and short pearson correlations. They become nearly the same at 40 nearest neighbors

The graph shows that when finding only the first five nearest neighbors, the two algorithms had only 40 percent of neighbors in common. However, when the number of nearest neighbors doubles to ten, the percentage of the same neighbors jumps to 70 percent. Then at 40 nearest neighbors the percent in common is in the upper 90s percentile and even continues to climb after this point. The similarity between the nearest neighbors of the two PCC algorithms shows that they are

nearly identical.

10.4 Cosine Similarity

Cosine similarity is another method to find nearest neighbors. Cosine similarity is mainly used for sparse data sets [5]. This is because the algorithm ignores the cases where neither user has data. This is the algorithm for cosine similarity:

$$cos(x, y) = \frac{\sum_{i=1}^{n}(A_i * B_i)}{\sqrt{\sum_{i=1}^{n}(A_i)^2}\sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

The $\sum_{i=1}^{n}(A_i * B_i)$ is the dot product between vector A and B. The denominator is the length of both A and B. The Cosine similarity algorithm gives a correlation between vector A and B [5]. The correlation ranges from 1, which is the same, to 0, which means the vectors are opposites.

11 Making Recommendations

11.1 Weighted Average

Once the nearest neighbors have been found, recommendations can then be made. One way to predict how much a user will like an item is by taking that user's nearest neighbors and finding the weighted average of those users' ranks for that item [16]. Weighted average is similar to regular average; however, some values add more than others to the overall average. In recommendation systems, the higher the correlation between the two users, the greater the effect will be on the user's average. Here is the weighted average formula:

$$Avg = \frac{\sum_{i=1}^{n}(w_i x_i)}{\sum_{i=1}^{n}(w_i)}$$

w_i is the weight. The weight in recommendation systems is based on the correlation between a user and the active user receiving the recommendation. This value could be equal to the correlation, or certain weights could be derived from the correlation [16]. Avg is the weighted average, and it is the predicted value for how the active user would rate the given item.

11.2 Filtering

Once predicted values for items have been made, it is time to filter out the bad choices and list the top recommendations. There are a few guidelines to adhere to however, when listing recommendations. These constraints include Pareto optimality, strategic manipulability, and independence of solutions. [8] Pareto optimality is fairly obvious and states that if everyone prefers item X to item Y, then X should be recommended over Y. This means that the more popular item should be recommended. The second guideline is strategic Manipulability, which is the idea

that a user can change his rankings to receive a recommendation on a certain item. This should not happen. The point of a recommendation system is to give users new ideas of things they may like, not to state opinions the user already knows. Preventing strategic manipulability is also important to stop fraud from occurring in the recommendations. [8] This means a company should not able to make rankings that would favor its own products over competitors' products. The third guideline is independence of solutions. This means that a recommendation system will recommend the next in line candidate if the first candidate is removed. [8]

12 Checking Prediction Accuracy

12.1 User Study

A user study consists of actual people using the recommendation. When conducting a user study, the recommendation system can be judged on how satisfied the user was with the recommendations, the time it takes to make recommendations, and how the tester likes using the recommendation system. [14] The downfall to user studies, however, is that they require money to pay the testers, as well as the necessity to have lots of time for them to use the system. There is a more practical way to test, though..

12.2 Offline approach

The offline approach to checking for recommendation accuracy consists of keeping part of the users' rankings hidden, then trying to check how accurate the recommendation system is at predicting those values. [15] The positive of testing this way is that it is free and does not require actual interaction with users. The ideal way to perform an offline test is to split the ratings data by time stamp. [15] The ratings that happened before the particular time will be the known values, and the ratings that hap- pened after will be the hidden values. This way it is sort of simulation of the past. There are a couple of ways to then judge the accuracy of these predictions.

12.3 Root Mean Squared Error

The root mean squared error (RMSE) method requires both the predicted value and the actual value of the ranking. Here is the formula:

$$A_{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(P_i - A_i)^2}{n}}$$

P_i is the predicted value for item i and A_i is the actual value of i. n is the total number of predicted values. The Focus of the RMSE is to really punish predictions that are off by a lot, [18] which is why the difference between the prediction and actually value is squared, which increases the average. The value returned by RMSE is always positive and the

closer it is to 0 the better the recommendations systems predictions are. RMSE would rank three predictions that were all off by one better than it would rank 2 correct predictions and one prediction that is off by three. Even the total that is off in both is three, the one that was off by a larger amount get rated worse. How close a recommendation is does matter, there is a big difference between a two and four star ranking, so thats why RMSE is the most commonly used algorithm for testing recommendation systems. [15]

12.4 Mean Absolute Error

Another formula for testing recommendations systems in Mean Absolute Error (MAE). MAE is just takes how much each prediction is off, and then averages these differences [15], so it doesn't matter if one number is off by three or if three numbers are off by one, they both will give the same answer. MAE is looks like this:

$$A_{MAE} = \frac{\sum_{i=1}^{n}(P_i - A_i)}{n}$$

P_i is the predicted ranking for item i and A_i is the actual ranking for item i. n is the total number of items whose values were being predicted.

13 My Project

For my demonstration, I used data from the online movie ranking system called MovieLens. This data consists of over 1 million rankings from about 6,000 users on almost 4,000 movies. I found the nearest neighbors by using both the Pearson coefficient and cosine similarity. I then used weighted averages to find the top recommendations. To test this system, I hid 10,000 ratings and made predictions for these hidden values and then checked how close the average of these predictions were to the actual values using both RMSE and MAE.

The following graphs show the MAE and RMSE on predictions on the 10,000 ratings that were hidden. Each graph is a comparison between the Pearson coefficient and the cosine similarity. The figure 2 is a comparison using the MAE to check the accuracy of the predictions. The figure 3 is a comparison using the RMSE. On both graphs the x axis shows the number of nearest neighbors found when making the prediction. The y axis is the value found from the MAE and RMSE in each graph.

Figure 2 shows that both the Pearson Correlation and the cosine similarity follow similar patterns of accuracy at different numbers of nearest neighbors. Both formulas are inaccurate at the extreme edges of nearest neighbors. They are most accurate in the middle at 1000 nearest neighbors. The Pearson Correlation is more accurate then the cosine similarity. The figure 3 shows similar patterns between the Pearson Correlation and the cosine

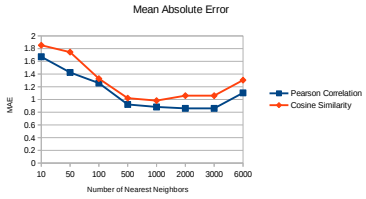

Figure 2: shows the MAE comparison between the pearson coefficient and cosine similarity

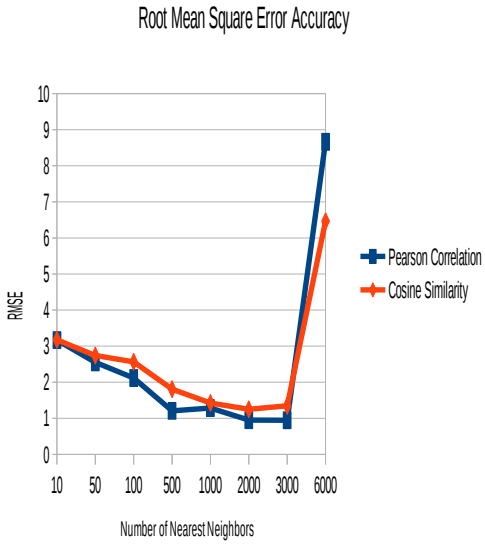

Figure 3: shows the RMSE comparison between the pearson coefficient and cosine similarity

similarity. The values are large at both ends of the total nearest neighbor spectrum. Both formulas are most accurate at 2000 nearest neighbors.

14 Results Conclusions

The Pearson correlation performed better than the cosine similarity when testing the results with both the MAE and RMSE. So it can be concluded that the cosine similarity is better to use when it comes to making predictions based off rankings. This is most likely due to the fact the Pearson correlation takes into account the different ways that people rank items, which is what pointed out in the Pearson Correlation section.

15 Future Trends

Successful recommendations can increase sales for companies, as well as make it more convenient for people to find things they would want or need. It is for these reasons that work is being done to recommendation systems to create more and more accurate predictions. In the near future, recommendation systems will be created to have the capability to predict exactly what someone will want before they know they want it. Recommendation systems will also be able to make recommendations for people in a timely manner so that wait time will no longer be a factor.

16 Increase Accuracy

16.1 Amazon

One of the most important goals for future recommendation systems is to increase the accuracy of recommendations. There is evidence of this desire for accuracy from many major companies. One such company is the e-commerce website Amazon, which is aiming to ship products before the customer actually orders them. [11] For this objective to be successful, Amazon will need to correctly predict who will want a particular item and when they will order it. If Amazon can successfully make accurate predictions, they will be to pull this off, which will ensure that consumers will receive their goods in a much shorter time. In order to realize their goal, Amazon will need to take into account more factors than only previous purchases and searches. An example of such a factor is the length of time a customer hovers his/her mouse over a selected item. [11] A longer hover over an item could indicate a greater desire for that particular item. It will take creative ideas like this, as well as many other similar ideas, to be able to create a recommendation system that is capable of making predictions with this degree of accuracy.

16.2 Netflix

Another example of a company looking to increase the accuracy of its predictions is Netflix. Netflix was willing to pay one million dollars for an algorithm that could predict ratings with 10 percent more accuracy than its current system. [6] This improvement had to be a 10 percent decrease in the Root Mean Square Error (RMSE) between the actual rankings and the predicted rankings. To increase the accuracy of these recommendations, the winning group took into account more variables than just the ratings of users. The most notable of these variables was the time stamp of the ratings. The group found that there was a distinction between ratings that happened in bulk at the same time, and ratings that happened individually. [17] The ratings that happened in bulk tended to be movies that users had watched in the past, so these movies tended to be rated on how the user remembered them as well as what the user considered that he/she had gotten out of viewing the movie. On the other hand, movies that were ranked individually tended to be recently viewed movies, which often resulted in a reaction to the movie based on emotions; therefore, these moves tended to be less accurate to the user's actual trends [17]. An example of this emotional reaction occurred for me immediately after I viewed Captain America: Winter Soldier. I felt the movie was very entertaining, and I probably would have ranked the movie fairly high if I had rated it soon after viewing it. However, after thinking about it for a period of time, I found that actually I had not liked the movie very much and so would have given it a much lower ranking. It is for this reason that movies that were ranked individually were given lesser weight than those that were ranked in bulk. Netfix is also taking into account the ranks of your friends to make recommendations [12] The logic behind this is that if a person's friend likes something, then that friend would recommend the particular movie. Obtaining information about and using others' friends is becoming possible today with the widespread use of social media, and through tricks like these, the accuracy of recommendation systems can be increased.

17 Increase Performance

To improve the accuracy of recommendation systems, more variables must be considered than just users' rankings and interests. Taking into account more variables requires more complex computations, thereby adding to the time required for the recommendation process to occur. However, it is actually a goal of recommendation systems to decrease the time it takes to make a recommendation for the user because a user is not going to wait very long to see the recommendations. It is for this reason that those who make recommendation systems try to create algorithms where the majority of the computations can be run when the user is offline. Basic user to item collaborative filtering methods have the capability to calculate the nearest neighbors of each user offline; however, in order to make the actual recommendations, the user must be online [11], or the system would have to

save the top list of recommendations for every user on the server and not update in real time. Amazon is an example of a company that does many of its recommendation computations offline [11]. It compares the similarities between two items offline, so then when the user is online, the system will already know items which are often purchased by the same user. In this way, recommendations can be made for the online user, based on these already known values. Doing the majority of the calculations offline speeds up the process of creating recommendations.

18 Conclusion

Many websites today implement Recommendation Systems for both the customers convenience and to make larger profits. There are a variety of different methods to develop the systems. Some are better then others in certain areas, such as the Pearson coefficient is more accurate than the cosine similarity when it comes to rating based collaborative filtering. Companies are seeing the value in recommendation systems are researching to increase the accuracy. Over time these systems have become better and will only continue to do so.

References

[1]

[2] J Bobadilla, F Ortega, A Hernando, and A Gutierrez. Recommender systems survey. *Knowledge-Based Systems*, 46:109–132, 2013. Cited References Count:253—151TO—ELSEVIER SCIENCE BV—PO BOX 211, 1000 AE AMSTERDAM, NETHERLANDS—Bobadilla, J.—Ortega, F.—Hernando, A.—Gutierrez, A.—ISI Document Delivery No.:151TO.

[3] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc.

[4] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[5] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[6] Eliot Buskirk. How the netflix prize was won, 2009.

[7] Z Huang, DD Zeng, and H Chen. Analyzing consumer-product graphs: Empirical findings and applications in recommender systems. *Management Science*, 53(7):1146–1164, 2007. Cited References Count:59—200FU—INFORMS—7240 PARKWAY DR, STE 310, HANOVER, MD 21076-1344 USA—Huang, Zan—Zeng, Daniel D.—Chen,

Hsinchun—ISI Document Delivery No.:200FU.

[8] J Iijima and S Ho. Common structure and properties of filtering systems. *Electronic Commerce Research and Applications*, 6(2):139–145, 2007. Cited References Count:12—225HU—ELSEVIER SCIENCE BV—PO BOX 211, 1000 AE AMSTERDAM, NETHERLANDS—Iijima, Junichi—Ho, Sho—ISI Document Delivery No.:225HU.

[9] KJ Kim and H Ahn. Collaborative filtering with a user-item matrix reduction technique. *International Journal of Electronic Commerce*, 16(1):107–128, 2011. Cited References Count:45—850ZX—M E SHARPE INC—80 BUSINESS PARK DR, ARMONK, NY 10504 USA—Kim, Kyoung-jae—Ahn, Hyunchul—ISI Document Delivery No.:850ZX.

[10] Joseph A Konstan and Riedl Riedl. Deconstructing recommender systems, 2012.

[11] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[12] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. Recommender systems with social regularization.

[13] P Ochi, S Rao, L Takayama, and C Nass. Predictors of user perceptions of web recommender systems: How the basis for generating experience and search product recommendations affects user responses. *International Journal of Human-Computer Studies*, 68(8):472–482, 2010. Cited References Count:38—614NU—ACADEMIC PRESS LTD- ELSEVIER SCIENCE LTD—24-28 OVAL RD, LONDON NW1 7DX, ENGLAND—Ochi, Paloma—Rao, Shailendra—Takayama, Leila—Nass, Clifford—ISI Document Delivery No.:614NU.

[14] YJ Park and KN Chang. Individual and group behavior-based customer profile model for personalized product recommendation. *Expert Systems With Applications*, 36(2):1932–1939, 2009. Cited References Count:32—390QE—PERGAMON-ELSEVIER SCIENCE LTD—THE BOULEVARD, LANGFORD LANE, KIDLINGTON, OXFORD OX5 1GB, ENGLAND—Park, You-Jin—Chang, Kun-Nyeong—ISI Document Delivery No.:390QE.

[15] Guy Shani and Asela Gunawardana. *Evaluating recommendation systems*, pages 257–297. Springer, 2011.

[16] Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. *Feature-weighted user model for recommender systems*, pages 97–106. Springer, 2007.

[17] Andreas Töscher, Michael Jahrer, and Robert M Bell. The bigchaos solution to the netflix grand prize. *Netflix prize documentation*, 2009.

[18] YP Ying, F Feinberg, and M Wedel. Leveraging missing ratings to improve online recommendation systems. *Journal of Marketing Research*, 43(3):355–365, 2006. Cited References Count:13—075IN—AMER MARKETING ASSOC—311S WACKER DR, STE 5800, CHICAGO, IL 60606-6629 USA—Ying, Yuanping—Feinberg, Fred—Wedel, Michel—ISI Document Delivery No.:075IN.

19 Reflection

One area of previous course work that I used to complete this project was the language Java. Many of the courses I have taken at Saint Johns required the use of the Java computing language. I chose to use Java because it is the language that I am most familiar with. It is not the most efficient language and efficiency is often important when it comes to recommendation systems. In my case though efficiency was not that important because the data set was not incredibly large and because the speeds at which the recommendations were made did not matter. With this project though, I had first hand eperience with how long it can take to do calculations in Java. Part of my project required running a data set over size 1,000,000 through an algorithm that was of complexity $O(n^2)$. It took quite a while to this computation.

Other skills that were useful in the completion of this course were technical writing and research. Both of these were used in the parallel computing course. During the course we wrote about the findings of our experiments in a technical we way. We also read several technical documents during the course, which were a very important skills for research seminar. Both these skills were sharpened during the course. During this course I have read more technical papers than I have previously over my entire life. Performing this many repetitions in a skill will naturally improve on it. The same is true for technical writing. I feel that during the semester I have learned say what I want to say in fewer words, which is an important aspect of technical writing.

As far as other skills from computer science that proved to be useful during the semester are work ethic and planning ahead. Computer science courses are not classes where work can done the night before. They require much planning and learning. This work ethic was essential when working on a project of this size. This also means then that these skills were enhanced during this experience.