Collaborative Filtering:

Implementation of a user-based recommendation system using K-Nearest Neighbor

Ranon Martin

Nova Southeastern University

Author's Note

This research project report was prepared for Artificial Intelligence (CISC 670), instructor: Wei Li, Ph.D in Fall 2015.

Prepared by Ranon Martin, Masters of Science in Computer Science program. Contact: rm1905@nova.edu.

Abstract

Collaborative filtering has been made popular through its use in recommendation systems, which plays an import role in the day-to-day operations of various Internet applications (Herlocker, Konstan, & Riedl, 2002; Formoso, Fernández, Cacheda, & Carneiro, 2005). Where it is used to increase sales for businesses and help customers find new products based on recommendations. The notion of collaborative filtering is driven by the rudimentary assumption that users' X and Y, existing within a given problem space, has similar ratings for n items, or exhibit similar behavior patterns, will rate or act similarly on other given items (Su & Khoshgoftaar, 2009). This research project reports the process and observations obtained from the implementation of a user-based collaborative filtering recommendation system based on the k-nearest neighbor algorithm, that was used to recommend one or more items to a particular user. The k-nearest neighbor algorithm plays a very import role in the operation of user-based collaborative filtering recommendation systems, to an extent where it affects its prediction accuracy.

Keywords: Collaborative filtering, K-nearest network, recommendation systems

Collaborative Filtering: Implementation of a user-based recommendation system using K-Nearest Neighbor

Collaborative filtering has been made popular through its use in recommendation systems, to predict a user's interest in an item based on the recommendations of other users who have similar interests (Herlocker, Konstan, & Riedl, 2002; Formoso, Fernández, Cacheda, & Carneiro, 2005). It stands out among other techniques for recommendation systems due to its reputation of providing good results, especially in the e-commerce domain (Formoso, Fernández, Cacheda, & Carneiro, 2005). Where it reaped a considerable amount of success through its use on websites such as Amazon.com and CDNow.com (Herlocker, Konstan, & Riedl, 2002).

At high-level, collaborative filtering bases its recommendations on information generated from the opinions and preferences of related users or subjects, instead of requirement descriptions (Herlocker J. L., 2000; Breese, Heckerman, & Kadie, 1998). Resulting in high quality recommendations, since they are generated from real users or subjects (Formoso, Fernández, Cacheda, & Carneiro, 2005).

Recommendation systems have proven its relevance, as it plays an important role in the day-to-day operations of various Internet applications (Formoso, Fernández, Cacheda, & Carneiro, 2005). Where it is used to make personalized suggestions to users based on their preferences, and help companies increase sales and profits by helping consumers to find desirable products (Wu, Joung, & Chiang, 2011; Formoso, Fernández, Cacheda, & Carneiro, 2005).

Neighborhood based algorithms, such as the k-nearest neighbor algorithm serves as the dominant approach that powers the predictive nature of collaborative filtering (Bell

& Koren, 2007). Providing the ability to make recommendations based on the preferences of *k* similar items or users (Formoso, Fernández, Cacheda, & Carneiro, 2005). K-nearest neighbor algorithm's simplicity and intuitive nature gives it the edge over the complex modern techniques that exist today to power collaborative filtering systems (Formoso, Fernández, Cacheda, & Carneiro, 2005).

The primary problem of this research project is to implement a user-based collaborative filtering recommendation system based on the k-nearest neighbor algorithm that will recommend one or more items to a particular user. A reliable training-testing dataset pair is ingested into the implemented collaborative recommendation, where a series experiments is carried out to demonstrate its predictive accuracy in relation to different parameters.

The background section looks into the background of collaborative filtering recommendation systems, describing the operation and techniques involved in a fully functional user-based collaborative filtering recommendation system. The challenges that exists in the use of user-based collaborative filtering for recommendation systems are also explored in the background section. The method section outlines the details of the implementation and associated experiments carried out on the implemented user-based collaborative filtering recommendation system, inclusive of the algorithm design, dataset characteristics, implementation hardware and software, and experiment design. Results produced from the experiments carried out on the implementation is presented in results section. Finally, in discussion section, the results are analyzed and the observations discussed.

Background

Overview

The notion of collaborative filtering is driven by the rudimentary assumption that a pair of given users, X and Y existing within a given problem space, has similar ratings for n items, or exhibit similar behavior patterns, will rate or act similarly on other given items (Su & Khoshgoftaar, 2009). Collaborative filtering involves the application of a series of predefined techniques to a dataset that consists of preferences for items by a set of users to predict or recommend additional items that the given user may have not tried as yet (Su & Khoshgoftaar, 2009).

Su and Khoshgoftaar (2009) went on to further explain collaborative filtering by outlining a typical scenario, where there is a problem space with a set of m users, $U = \{u_1, u_2, \ldots u_m\}$, and a set of n items, $I = \{i_1, i_2, \ldots\ i_n\}$, where each user, u_i, has a list of associated items, Iu_i, rated by the user. Item ratings can be either be an indication based on a numeric scale of one-to-five (such as a five-star rating system), or an implicit indication such as purchases (Su & Khoshgoftaar, 2009). A rating is typically represented by a value that exists within the defined numeric scale or a Boolean value (Kelleher & Bridge, 2004). Users' preferences are standardly represented by a $n \times m$ user-item matrix with the ratings, r (Kelleher & Bridge, 2004). An example of a user-item matrix for a set of movies is illustrated by Table 1. Where $r_{u,i}$, a rating of an item i by a user u, is represented by a numeric rating on the scale 1 – 5, or ? if it is not rated. It is the norm for user-item matrix to be sparse, when performing collaborative filtering, since a small percentage of the total number item have been rated by each user (Herlocker, Konstan, & Riedl, 2002).

	Reservoir dogs	Naked gun	Aliens	Fargo	Star wars	Taxi driver
Ann	3	?	?	4	4	4
Bob	?	2	5	5	2	?
Col	3	5	?	3	3	3
Deb	?	?	?	?	3	?
Edd	5	4	2	4	3	3
Flo	5	?	4	?	?	?

Table 1. An example user-item matrix depicting users' ratings for a set of movies.

The term active user, $u_a \in U$, is used to refer to the user who is interacting with the recommendation system, or is the subject of a recommendation (Kelleher & Bridge, 2004). According to Kelleher and Bridge (2004), the collaborative filtering recommendation system computes the predicted rating, $p_{u_a,i}$, for the active user, u_a, and an item i by using the r for $r_{u_a,i} = ?$. Based on an example from Table 1, if Col is the active user, then a predication can be made for the movie "Aliens", which hasn't been rated by the Col.

User-based collaborative filtering recommendation system, which is one of the two memory-based collaborative filtering recommendation system, uses a subset or the entire user-item matrix to generate a prediction (Su & Khoshgoftaar, 2009; Kelleher & Bridge, 2004). The sub-set of appropriate users, referred to as neighbors, are chosen based on their degree of similarity to the active user (Su & Khoshgoftaar, 2009; Herlocker, Konstan, & Riedl, 2002). If the task is to generate a Top-N recommendation, then the k most similar users (nearest neighbors) are extracted as the subset (Su & Khoshgoftaar, 2009). A weighted aggregate of the neighbors' ratings is then used to generate predictions for the active user (Herlocker, Konstan, & Riedl, 2002).

For example, from table 1, Col's rating for the movie "Aliens" can be predicted based on Edd's opinion, since he is Col's best neighbor. From observing the table, it can be seen that Col and Edd agreed closely with all the movies that they have both rated. On the other hand, Bob is not such a good neighbor, and will have a lesser impact on Col's prediction than Edd.

Operation

Generally, the operation of a user-based collaborative filtering recommendation system can be broken down into three steps (Herlocker, Konstan, & Riedl, 2002):

1. Perform similarity computation, to find the similarity between the active user and each of the other users, by way of distance, correlation or weight (Su & Khoshgoftaar, 2009; Herlocker, Konstan, & Riedl, 2002).
2. Select the k - nearest neighbors, a subset of users with the highest similarity, to use as a set of predictors (Herlocker, Konstan, & Riedl, 2002).
3. Perform prediction computation, by normalizing and computing a weighted average of the selected neighbors' ratings (Herlocker, Konstan, & Riedl, 2002).

Techniques

Similarity computation. According to Su and Khoshgoftaar (2009), similarity computation is a crucial step in the operation of a memory-based collaborative filtering recommendation system. Where the similarity computation is used in user-based collaborative filtering to calculate the similarity $w_{u,v}$, between the active user and another user who have rated the same items.

Euclidean distance-based similarity. The Euclidean distance-based similarity measure is based on the principle of calculating the straight-line distance between two points, which when applied in collaborative filtering, calculates the similarity distance between two users (Kumar, Gupta, Singh, & Shukla, 2015). The similarity $w_{u,v}$, between users u and v with co-rated items I, can be calculated by the Euclidean distance formula:

$$w_{u,v} = \sqrt{\sum_{i \in I} \left(r_{u,i} - r_{v,i}\right)^2}$$

Kumar, Gupta, Singh and Shukla (2015) has found that Euclidean distance-based similarity is out-performed by the Pearson correlation-based similarity measure in cases where the dataset used is very large, not normalized well and represents a very diverse set of users' interests.

Pearson correlation-based similarity. The Pearson correlation-based similarity computation measure is based on Pearson correlation, which measures the linear relationship of two variables (Su & Khoshgoftaar, 2009). To the calculate the similarity $w_{u,v}$, between users u and v, when performing user-based collaborative filtering, the following formula is used:

$$w_{u,v} = \frac{\sum_{i \in I}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I}\left(r_{u,i} - \bar{r}_u\right)^2}\sqrt{\sum_{i \in I}\left(r_{v,i} - \bar{r}_v\right)^2}}$$

Where I are the items that are rated by both users and $\bar{r}_u$ is the average rating for the uth user's co-rated items (Su & Khoshgoftaar, 2009).

Vector cosine-based similarity. The vector cosine-based similarity computation measure is based on the cosine similarity formalism, which when adopted in user-based collaborative filtering, measures the similarity between two users by treating each user as a vector of ratings and calculating the cosine of the angled formed by the ratings vectors (Su & Khoshgoftaar, 2009). The following formula calculates the similarity $w_{u,v}$, between users u and v, when performing user-based collaborative filtering:

$$w_{u,v} = \frac{\sum_{i \in I_{uv}} r_{u,i} r_{v,i}}{\sqrt{\sum_{i \in I_u} \left(r_{u,i}\right)^2} \sqrt{\sum_{i \in I_v} (r_{v,i})^2}}$$

Where I_{uv} are items that rated by both users and $r_{u,i}$ is the rating of the item i by user u (Wu, Dept. of Comput. Sci., He, Ren, & Xia, 2008). Breese, Heckerman, and Kadie (1998) found that vector cosine-based similarity is out performed by Pearson correlation-base similarity when used in collaborative filtering.

Prediction computation. Prediction computation is the paramount step in the operation of the user-based collaborative filtering recommendation system, as this is the step that does the heavy lifting to obtain the actual predictions or recommendations based on the subset of nearest neighbors of the active user (Su & Khoshgoftaar, 2009). According to Su and Khoshgoftaar (2009), the prediction computation is carried out by obtaining a weighted sum of each member that exists in the subset of nearest neighbors of the active user. A prediction $P_{a,i}$ for the active user a, for an item i, can by computed by the formula (Su & Khoshgoftaar, 2009):

$$P_{a,i} = \overline{r_a} + \frac{\sum_{u \in U} (r_{u,i} - \bar{r}_u) \times w_{a,u}}{\sum_{u \in U} |w_{a,u}|}$$

Where $u \in U$ represents all the users who have rated i, $w_{a,u}$ is the similarity weight between user a and user u, and $\bar{r}_a$ and $\bar{r}_u$ represents the average ratings for users a and u respectively for all the co-rated items (Su & Khoshgoftaar, 2009).

Characteristics and challenges. Adomavicius & Tuzhilin (2005) states that collaborative filtering recommendation systems can deal with any kind of content and make recommendations for any items, due to its underlying notion of using other user's recommendations. Although collaborative filtering recommendation systems pose such a powerful characteristic in addition to it being easy to implement, scale well to co-rated items and the ability for new data to be added easily and incrementally, they have their own share of challenges (Adomavicius & Tuzhilin, 2005; Su & Khoshgoftaar, 2009).

Data sparsity. The data sparsity challenge occurs in situations where the user-item matrix of a large dataset, used for collaborative filtering is extremely sparse, leading to the decline in the performance of the predictions or recommendations produced (Su & Khoshgoftaar, 2009). Su & Khoshgoftaar (2009) also states that this challenge is present in scenarios where there is insufficient data to be used to find similar users, especially when attempting to make recommendations for a new user, who has no rating. The addition of new items, that have no rating, is also popular scenario (Adomavicius & Tuzhilin, 2005).

Scalability. The scalability challenge occurs where the number of users and items ingested by the collaborative filtering recommendation system increase tremendously, resulting in traditional collaborative filtering algorithms suffering scalability issues, which requires computational resources extending beyond practical levels (Su & Khoshgoftaar, 2009).

Gray Sheep. Gray sheep is a term used to refer to users who has opinions having a consistent characteristic of not aligning with any group of users in the user-item matrix, which creates a difficulty in making recommendations or predictions for such a user (Su & Khoshgoftaar, 2009). As of such, Su & Khoshgoftaar (2009) went on to state that gray sheep don't benefit from collaborative filtering.

Shilling attacks. Shilling attacks are cases where a user or a set of users may maliciously provide tons of positive recommendations in favor of a particular item or tons of negative recommendations for a competitor's item to manipulate the collaborative filtering recommendation system's operation (Su & Khoshgoftaar, 2009).

Synonymy. The synonymy challenge is present in a problem space where there are a number of the same or similar items with different names, being treated as separate items, because of the inability of the recommendation system to create the requisite associations (Su & Khoshgoftaar, 2009). Su & Khoshgoftaar (2009) also noted that the prevalence of synonyms within a user-item matrix decreases the recommendation or prediction performance of collaboration recommendation systems.

Diversity and long tail. The main motivation behind collaborative filtering recommendation systems is to help users to discover new items, and in return increase diversity (Fleder, 2009). however, Felder (2009) went on to state that there is a school of thought, that many recommendation system designs tend to push users more towards items that are popular already, hence, acting in opposition of the indent and reducing diversity.

Method

Overview

Knowledge discovery in databases (KDD) process was applied to the implemented user-based collaborative filtering recommendation system, as explained in Dunham (2002), to extract useful information from the dataset. The KDD process consists of the following steps:

1. Selection: The dataset was obtained from the source of source of choice.
2. Preprocessing: The dataset retrieved was preprocessed so no work had to be done to correct or remove erroneous data and supply or predict missing data. However, the sub-dataset of choice was extracted from the group of datasets obtained.
3. Transformation: Both training and testing datasets were transformed into separate user-item matrix as required by the collaborative filtering recommendation system.
4. Data mining: Both the training and testing datasets were applied to the implemented user-based recommendation system to obtain one or more recommendations for the active user. Performance data were also extracted from the recommendation system for evaluation purposes.
5. Interpretation: The results obtained for the data mining operation were presented in a useful manner.

Dataset

The dataset used for this research project was extracted from the MovieLens data set, collected by the Grouplens research project at the University of Minnesota

(GroupLens Research Lab, 1998). This dataset consists of 100, 000 ratings from 943 anonymous users on 1682 movies. Each of the users rated at least 20 movies, with a score on the scale of 1-5. The data set also has simple demographic information such as age, gender, occupation and zip code for each user.

The user-rating dataset consists of user id, item id, rating and timestamp attributes. The user-rating dataset was split into disjoint training and testing datasets with an 80% 20% distribution, respectively. Where the testing dataset has exactly 10 ratings per user.

Algorithm

To implement the user-based collaborative filtering recommendation system that will recommend one or more movies to the active user, collaborative filtering steps were implemented in the design of the recommendation system to predict the rating for each of the items that have not been rated by the active user, as illustrated in figure 1. Each unrated item prediction rating is then ranked, where only the Top-N ranked items will be served to the active user as recommendations.

INPUT:

- The id for the active user, *active_user*
- The number of recommendations to return, *item_count* : initialized to 1
- The user-item matrix, *user_item_matrix*

OUTPUT:

An array of recommendations

PROCEDURE:

```
INITIALIZE:
    • predictions = []
    • dividends, divisors = {}, {}

co_rated_items = get_co_rated_items(active_user, user_item_matrix)
active_user_average_rating = compute_average_rating(active_user, co_rated_items)

#Extract k-nearest neighbors
neighbors = similar_users(active_user)

FOR EACH neighbor, similarity IN neighbors

  CONTINUE; IF similarity <= 0

  ratings = user_item_matrix[neighbor]

  FOR EACH item, rating IN ratings
    # Only make predictions for items not rated by active user
    IF user_item_matrix[active_user][item] == NIL
      INITIALIZE: dividends[item] = 0; IF dividends[item] == NIL
      INITIALIZE: divisors[item] = 0; IF dividends[item] == NIL
      average_co_rating = average_rating(neighbor, co_rated_items)

      dividends[item] += (rating - average_co_rating) * similarity
      divisors[item] += similarity
    END
  END FOR
END FOR

# Compute predictions
FOR EACH item, dividend IN dividends
  predictions << { "item" => item, "predicted_rating" => active_user_average_rating + (dividend / divisors[item])}
END FOR

# Rank predictions by predicted rating
ranked_predictions = rank_by_rating(predictions)

# Return the top-N recommendations
RETURN ranked_predictions[1...item_count]
```

Figure 1. Recommendations procedure

Similarity computation was implemented with the Euclidean distance function, illustrated in figure 2, and Pearson correlation function, illustrated in figure 3, enabling the evaluation of the impact of each similarity measuring function on the prediction accuracy of the recommendation system.

```
INPUT:
    •   The id for the active user, active_user
    •   The id for the user that is being compared to the active_user, other_user
    •   The user-item matrix, user_item_matrix

OUTPUT:
The similarity score: FLOAT

PROCEDURE:
INITIALIZE:
  co_rated_items = NIL
  sum_of_squares = 0

co_rated_items = get_co_rated_items(active_user, other_user)

# No items rated in common
RETURN 0 IF co_rated_items == NIL

# Find euclidean distance
FOR EACH item IN co_rated_items
  active_user_rating = user_item_matrix[active_user][item]
  other_user_rating = user_item_matrix[other_user][item]
  sum_of_squares += square(active_user_rating - other_user_rating)
END FOR
RETURN 1/(1 + sqrt(sum_of_squares))
```

Figure 2. Computing the similarity score of two users based on the Euclidean distance.

```
INPUT:
```

- The id for the active user, *active_user*
- The id for the user that is being compared to the active_user, *other_user*
- The user-item matrix, *user_item_matrix*

OUTPUT:

Similarity score: FLOAT

PROCEDURE:

```
INITIALIZE:
  co_rated_items = []
  dividend = 0
  multiplicand = 0
  multiplier = 0

co_rated_items = get_co_rated_items(active_user, other_user)

# No items rated in common
RETURN 0 IF co_rated_items == NIL

user_average_rating = average_rating(active_user)
other_user_average_rating = average_rating(other_user)

FOR EACH item IN co_rated_items
  active_user_rating = user_item_matrix[active_user][item]
  other_user_rating = user_item_matrix[other_user][item]

  dividend += (active_user_rating - user_average_rating) * (other_user_rating - other_user_average_rating)
  multiplicand += (active_user_rating - user_average_rating) ** 2
  multiplier += (other_user_rating - other_user_average_rating) ** 2
```

```
END FOR

divisor = sqrt(multiplicand) * sqrt(multiplier)

IF divisor == 0
  RETURN 0
ELSE
  RETURN dividend/divisor
END IF
```

Figure 3. Computing the similarity score of two users based on Pearson correlation.

The weighted average of the k nearest neighbors' ratings are computed to obtain the predicted rating for a particular item, as outlined in figure 1. The active user's predicted ratings for the item that he has not rated was ranked, and the Top-N items were presented as recommendations, as in figure 1.

Implementation

The user-based collaborative filtering recommendation systems was implemented using version 2.2.1 of the Ruby programming language. On an Apple MacBook Pro running a 2.4 GHz Intel Core i5 processor with 8 GB DDR3 RAM operating at 1600 MHz. The implemented recommendation system was tested with the version 2.6 build 361 of the Terminal command line interface tool on OS X El Capitan version 10.11.1.

Evaluation method

Root mean squared error (RMSE) was used as the prediction error evaluation metric to measure the accuracy of the predictions made by the implemented user-based collaborative filtering recommendation system. Where the difference between the predicted rating and actual rating were computed for each prediction. Cacheda (2011) made mention of the surge in the popularity of the usage of RMSE as an error evaluation metric in recent years due to its use in the Netflix Prize competition.

The root mean squared error was computed by the formula:

$$RMSE = \sqrt{\frac{1}{|S_{test}|} \sum_{u,i \in S_{test}} \left(P_{u,i} - r_{u,i}\right)^2}$$

Where S_{test} is the set of all user ratings that exists in the testing dataset, $P_{u,i}$ is the predicted rating for item i, by user u, and $r_{u,i}$is the actual rating for the item by said user, existing in the testing dataset.

The accuracy of the predictions was measured by running the user-based collaborative filtering recommendation system to recommend a movie to an active user in different scenarios where the neighborhood size was incremented by 50 users each time. This operation was carried out with each of the implemented similarity measuring algorithms, Euclidean distance and Pearson correlation similarity measures, to evaluate the effect of each on the prediction accuracy.

Results

Table 2 presents the data that were obtained after running the implemented user-based collaborative filtering recommendation system, with the Euclidean similarity function and different neighborhood sizes, to recommend a movie for the user with id “1”.

Neighborhood size	**Recommended movie**	**RMSE**
100	Thin Line Between Love and Hate, A (1996)	1.441821861
150	Thin Line Between Love and Hate, A (1996)	1.370998051
200	Thin Line Between Love and Hate, A (1996)	1.214586215

250	Thin Line Between Love and Hate, A (1996)	1.637985158
300	Thin Line Between Love and Hate, A (1996)	1.635368186
350	Thin Line Between Love and Hate, A (1996)	1.496132487
400	Thin Line Between Love and Hate, A (1996)	1.241180042
450	Thin Line Between Love and Hate, A (1996)	1.005767862
500	Thin Line Between Love and Hate, A (1996)	0.9943542
550	Thin Line Between Love and Hate, A (1996)	1.020226401
600	Thin Line Between Love and Hate, A (1996)	0.972955831
650	Lamerica (1994)	0.975725467
700	Lamerica (1994)	0.937030365
750	Lamerica (1994)	0.914358487
800	Lamerica (1994)	0.878612843
850	Lamerica (1994)	0.857339652
900	Lamerica (1994)	0.859377041
950	Aiqing wansui (1994)	0.853784364

Table 2. Prediction accuracy measurements when using the Euclidean similarity function.

Table 3 presents the data that were obtained after running the implemented user-based collaborative filtering recommendation system, with the Pearson correlation similarity function and different neighborhood sizes, to recommend a movie for the user with id “1”.

Neighborhood size	**Recommended movie**	**RMSE**
100	Thin Line Between Love and Hate, A (1996)	0.833386634
150	Bogus (1996)	0.639375393
200	Thin Line Between Love and Hate, A (1996)	0.652627712
250	Dunston Checks In (1996)	0.708508923

300	Dunston Checks In (1996)	0.724779648
350	Great Day in Harlem, A (1994)	0.719495489
400	Great Day in Harlem, A (1994)	0.715778212
450	Great Day in Harlem, A (1994)	0.73101888
500	Great Day in Harlem, A (1994)	0.740593196
550	Great Day in Harlem, A (1994)	0.759764781
600	Boys, Les (1997)	0.771718333
650	Boys, Les (1997)	0.773734115
700	Boys, Les (1997)	0.780546591
750	Aiqing wansui (1994)	0.783015445
800	Aiqing wansui (1994)	0.78290333
850	Aiqing wansui (1994)	0.78290333
900	Aiqing wansui (1994)	0.78290333
950	Aiqing wansui (1994)	0.78290333

Table 3. Prediction accuracy measurements when using the Pearson correlation similarity function.

Figure 4 provides a plot of the data presented in table 2 and table 3, to illustrate the impact that each similarity measure has on the prediction accuracy of the implemented user-based collaborative filtering recommendation system.

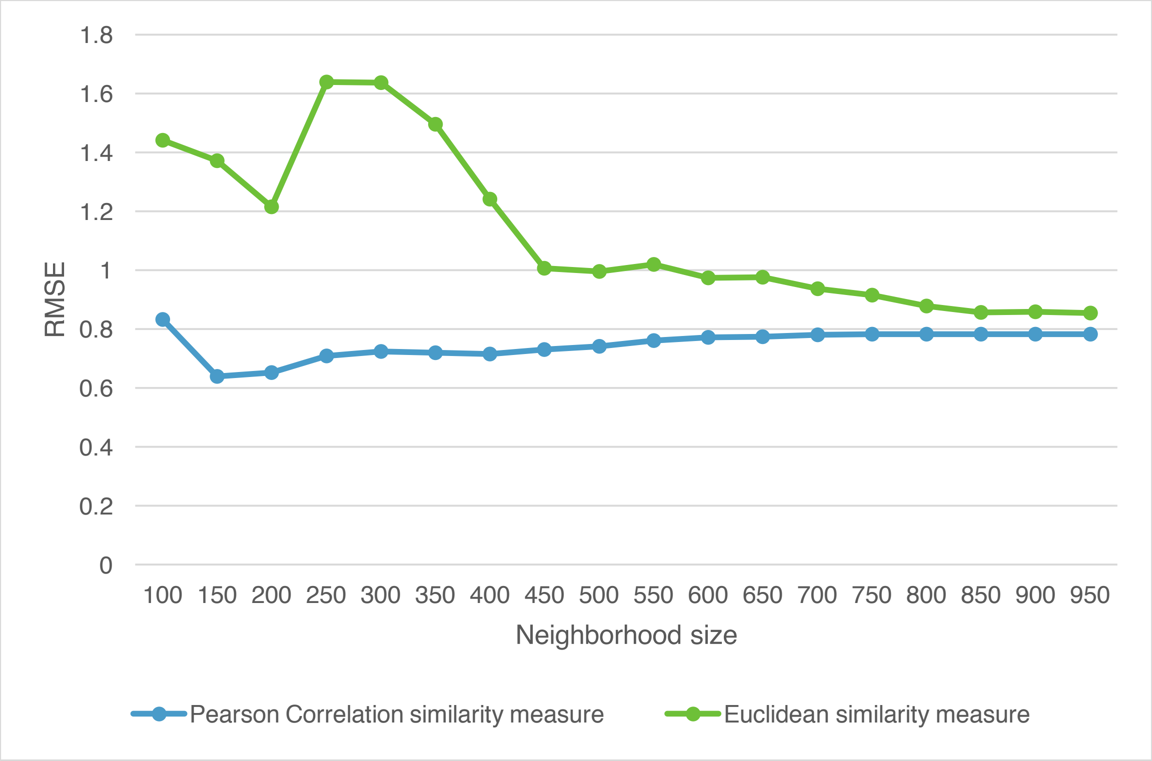

Figure 4. Comparison of the similarity functions on the prediction accuracy over a range of neighborhood sizes.

Discussion

After running the implemented user-based collaborative filtering recommendation system under different circumstances, some very important observations were noted. Through observation of the data obtain from both sets of experiments, where predictions were carried out using the Euclidean distance and Pearson correlation similarity measures, the prediction accuracy of the recommendation system improved with the increase of the neighborhood size. However, the Euclidean distance similarity measure displayed better improvement over the increase of the neighborhood size, when compared to the performance of Pearson correlation's improvement. Both sets of experiments started out with a neighborhood of 100 users, after closing out each experiment with a neighborhood of 950 users, the experiment using the Euclidean distance similarity measure enjoyed a 40.8 percent improvement in prediction accuracy, while the experiment using the Pearson correlation similarity measure displayed only 6.1 percent in prediction accuracy. Looking at the data from a different angle, we can conclude that the Pearson correlation similarity has a better prediction accuracy, as the final root mean squared error for that experiment is 8.3 percent lower than the experiment using Euclidean similarity measure, and also experienced its best prediction accuracy at the point where the neighborhood size was 150 users.

We can also observe that, as the prediction accuracy of the user-based collaborative filtering recommendation system improves, its recommendation changes, to a more accurate recommendation. This can be seen in table 2, where the neighborhood size was increased from 900 to 950, the accuracy of the prediction improved by only 0.65 percent, resulting in a change of the recommended movie.

From the experiments carried out on the implemented user-based collaborative filtering collaborative recommendation system, we can learn that the K-nearest neighbor approach plays a major role in its operation. As increasing the neighborhood can result in significant improvement in the prediction accuracy, as in the case of the set of experiments using Euclidean distance, or result in a reduction of accuracy, as in the case of the set of experiments using Pearson correlation. This may result from the thinking that not all neighbor ratings beyond a particular neighborhood size might be equally valuable to the prediction.

References

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 734 - 749.

Bell, R. M., & Koren, Y. (2007). Improved Neighborhood-based Collaborative Filtering. *KDD Cup and Workshop 2007* (pp. 7-14). San Jose, California, USA: ACM.

Breese, J. S., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. *UAI'98 Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence* (pp. 43-52). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Cacheda, F. C. (2011). Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transactions on the Web (TWEB), 5*(1), 2.

Dunham, M. H. (2002). *Data mining introductory and advanced topics.* Upper saddle river, NJ, USA: Pearson Education, Inc.

Fleder, D. &. (2009). Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Management science, 55*(5), 697-712.

Formoso, V., Fernández, D., Cacheda, F., & Carneiro, V. (2005, July). Distributed architecture for k-nearest neighbors recommender systems. *World Wide Web, 18*(4), 997-1017.

GroupLens Research Lab. (1998, April 1). *MovieLens 100K Dataset*. Retrieved October 1, 2015, from Grouplens: http://grouplens.org/datasets/movielens/100k/

Herlocker, J. L. (2000). *Understanding and improving automated collaborative filtering systems.* University of Minnesota. Minnesota, USA: University of Minnesota.

Herlocker, J., Konstan, J. A., & Riedl, J. (2002, october). An Empirical Analysis of Design Choices in Neighborhood-Based Collaborative Filtering Algorithms. *Information Retrieval Journal, 5*(4), 287 - 310.

Kelleher, J., & Bridge, D. (2004, June). An Accurate and Scalable Collaborative Recommender. *Artificial Intelligence Review, 21*(3), 193-213.

Kumar, A., Gupta, S., Singh, S., & Shukla, K. (2015). Comparison of various metrics used in collaborative filtering for recommendation system. *Contemporary Computing (IC3), 2015 Eighth International Conference on* (pp. 150 - 154). Washington, DC: IEEE.

Su, X., & Khoshgoftaar, T. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 4.

Wu, F., Dept. of Comput. Sci., E. C., He, L., Ren, L., & Xia, W. (2008). An effective similarity measure for collaborative filtering. *Granular Computing, 2008. GrC 2008. IEEE International Conference on* (pp. 659 - 664). Washington DC: IEEE.

Wu, L.-L., Joung, Y.-J., & Chiang, T.-E. (2011). Recommendation Systems and Sales Concentration: The Moderating Effects of Consumers' Product Awareness and Acceptance to Recommendations. *HICSS '11 Proceedings of the 2011 44th Hawaii International Conference on System Sciences* (pp. 1-10). Washington, DC, USA: IEEE Computer Society.