Submitted to *INFORMS Journal on Computing*
manuscript JOC-2017-01-OA-008

Understanding the Impact of Individual Users' Rating Characteristics on Predictive Accuracy of Recommender Systems

Xiaoye Cheng, Jingjing Zhang, and Lu (Lucy) Yan
Department of Operations and Decision Technologies, Kelley School of Business, Indiana University
{xc37, jjzhang, yanlucy}@indiana.edu

In this study, we investigate how individual users' rating characteristics affect the user-level performance of recommendation algorithms. We measure users' rating characteristics from three perspectives: rating value, rating structure and neighborhood network embeddedness. We study how these three categories of measures influence the predictive accuracy of popular recommendation algorithms for each user. Our experiments use five real-world datasets with varying characteristics. For each individual user, we estimate the predictive accuracy of three recommendation algorithms. We then apply regression-based models to uncover the relationships between rating characteristics and recommendation performance at the individual user level. Our experimental results show consistent and significant effects of several rating measures on recommendation accuracy. Understanding how rating characteristics affect the recommendation performance at the individual user level has practical implications for the design of recommender systems.

Key words: recommender systems; predictive accuracy; rating characteristics; rating value; rating structure; network embeddedness

1. Introduction and Motivation

Recommender systems assist users in content discovery and exploration by suggesting items that match users' personal preferences. Recommender systems rely on users' preference data (e.g., users' prior purchases, browsing history, submitted ratings) to model and estimate users' preferences for unconsumed items. The outcome of a typical recommender system is a list of suggested items with highly predicted ratings. Prior studies in the recommender systems literature have focused primarily on the overall recommendation performance for all users (usually measured by the overall accuracy of rating predictions on the entire dataset), and researchers have proposed numerous novel algorithms to improve aggregate population-level recommendation performance (e.g., Bell and Koren 2007, Deshpande and Karypis 2004, Funk 2006, Koren et al. 2009, Resnick et al. 1994, Sarwar et al. 2001). Little research, however, has focused on recommendation performance at the individual-user level and how user-level performance is affected by each user's rating characteristics.

User-level recommendation performance is directly observable in individual users and has profound effects on users' experiences with the recommender system. A system's overall performance for the entire user population, however, does not always reflect the quality of recommendations received by each user. Even in a system with high aggregate performance, some users may receive low-quality recommendations and, thus, have an unpleasant experience with the system. For example, Netflix has put significant effort into improving the accuracy of its recommender system, including incorporating many high-performing recommendation algorithms and hosting a competition, called "Netflix Prize," in which one million dollars was offered to the team that could improve the overall accuracy of Netflix's algorithm by 10% (Bennett et al. 2007). Despite improvements in overall performance, there is still great variation in individual users' experiences with Netflix's recommendation service. Although many users find their recommendations to be accurate, some users continue to receive low-quality recommendations, and find them inaccurate and irrelevant (Allen 2017). Similarly, users of other real-world recommender systems (including the successful ones at companies such as Amazon and Pandora) also have reported differing degrees of satisfaction with the same systems. As can be seen, improvement of the overall recommendation performance for all users does not always lead to improvement for every user.

Recent research has started to recognize the importance of user-centric evaluation and the need to study user-level recommendation performance (e.g., Pu et al. 2012). For an individual user, the quality of a system's recommendations for him or her (as opposed to the overall recommendation quality for all users), has a direct impact on the user's acceptance of recommendations as well as his or her willingness to continue interacting with the system. In the following, we use a simple numeric example to illustrate different and even contrasting experiences based on population-level (i.e., the dataset of all users), and individual-level (i.e., a particular user), recommendation performance.

Suppose we have a movie recommender system with five users and five movies. Users provide their ratings on movies using a 5-star rating scale. All of the ratings of the system can be represented by a partially filled rating matrix, as depicted in Table 1. To calculate the predictive accuracy, we apply the standard holdout cross-validation approach: We randomly reserve a number of ratings (e.g., two ratings by Alex on "Star Wars" and "Minions" and one rating by Bob on "Star Wars") in the test set and use the remaining ratings as training data. The recommender system calculates the similarity of scores between Alex and all other users, for example, based on a user-based collaborative filtering (CF) algorithm (Konstan et al. 1997), and predicts that Alex would rate "Star Wars" and "Minions" at 2.32 and 2.95 stars, respectively. Similarly, the system uses the same algorithm on Bob, and predicts that Bob would rate "Star Wars" at 2.34 stars. Overall, the mean absolute error (MAE) of predictions on the overall evaluation set is 0.69. At the individual user level, however, there is big difference between the prediction accuracy for Alex and Bob. For Bob, the predicted rating of 2.34 is very close to his actual rating (2 stars). Yet, the predicted rating for Alex on "Star Wars" is much different from his actual rating. The mean absolute prediction error for Alex (0.865) is much larger than that for Bob (0.34). Although Alex and Bob provided the same number of ratings, the same recommendation algorithm resulted in different accuracy performance for the two users. In this research, we are interested in understanding what leads to such differences in user-level recommendation performance.

Table 1 Movie Recommendation Example.

User	Kung Fu Panda	Spider Man	Harry Potter	Star Wars	Minions
Alex	5	3	4	**4**	**3**
Bob	4	5	4	**2**	
Claire		3	5	2	1
David	4	4	3	2	3
Emily	3	5	4	3	5

Thus, the goal of this study is to investigate which rating characteristics of an individual user affect recommendation performance for the user. To this end, we build an explanatory model to examine the direction and significance of the influence of a user's rating characteristics on user-level predictive accuracy of recommendation algorithms. Specifically, we measure each user's rating characteristics in regard to three aspects: rating value, rating structure, and neighborhood network embeddedness. Using three popular recommendation algorithms on five rating datasets, our results show that the effects of these rating characteristics are significant and consistent with regard to the accuracy performance across different algorithms and datasets. This research provides insight into which rating characteristics play a more (or less) important role and have a positive (or negative) influence on user-level accuracy performance of recommendation algorithms.

This research makes several contributions to the recommender systems field. First, we look at recommendation performance at the individual-user level, as opposed to the dataset level. The focus of our analysis is at a granularity level different from that of prior research (e.g., Adomavicius and Zhang 2012) and, therefore, offers new insight. Second, we present an approach to construct an implicit network based on users' co-rating patterns and demonstrate the value of using an implicit network to predict accuracy performance. Prior studies have focused primarily on explicit networks based on self-indicated social relations (e.g., Facebook friends, Twitter followers), while the implicit network has been overlooked. Our work, thus, introduces and demonstrates the value of the implicit network in the recommender systems context. Third, our findings provide many practical implications for the design and implementation of recommender systems. Later in this paper we provide several example approaches to demonstrate how recommender system designers can use our model to customize the system design for individual users.

2. Related Work

2.1. Data Distribution in Recommender Systems

Recommender systems rely on users' feedback on consumed items to generate recommendations (e.g., Berkovsky and Freyne 2015, Jonnalagedda and Gauch 2013). The input of recommendation algorithms includes both the feedback provided explicitly by users (such as ratings or likes) and the feedback that can be collected implicitly by monitoring users' behaviors (such as clicking or browsing patterns). Implicit feedback represents inferences about users' interests based on their actions observed by the system. For example, in the music recommender system Last.fm (`www.last.fm`), if a user listens to a track five times, the system infers that the user has an interest in that track (Jawaheer et al. 2010). Many algorithms have been developed to utilize implicit feedback in building user models (e.g.,Hu et al. 2008, Li and Chen 2016), and some of these algorithms even allow users to indicate which feedback data they would like the algorithm to use to generate recommendations (Miller et al. 2004). A key drawback of using implicit feedback is that it is difficult to identify missing values as either negative (e.g., items disliked by user), or positive (e.g., relevant items but unseen by users) (e.g., Pan et al. 2008). This inherent property of implicit feedback leads to challenges in proper evaluation of the recommendation quality.

In contrast, explicit feedback provided by users in the forms of ratings can be positive or negative. For example, Amazon asks users to rate the items that they have purchased on a 5-star scale (with 1 as "I hate it" and 5 as "I love it"). In representing the user's interests, it is generally recognized in literature that explicit feedback is more accurate than implicit feedback (e.g., Jawaheer et al. 2010). Therefore, in the recommender systems literature, the most common approach to modeling users' preferences for items is via explicit feedback, such as numeric ratings (Adomavicius and Tuzhilin 2005). Users' ratings on consumed items can be used as inputs to the recommender systems. In real-world recommender systems, because users consume and rate only a small set of items, users' ratings are sparse (e.g., Herlocker et al. 2000). The goal of a recommender system is to estimate ratings for the items that have not been consumed by a user, and then recommend the item(s) with the highest estimated rating(s) to the user. Our research focuses on this most common setting of recommender systems and examines how the performance of recommendation algorithms is influenced by the characteristics of the explicit ratings provided by users.

Prior literature has shown that the characteristics of rating data have a significant influence on the performance of recommender systems (e.g., Adomavicius and Zhang 2012, Hsu et al. 2004). Such influences are algorithm-specific, that is, an algorithm may demonstrate better performance on some datasets than on others (Kluver and Konstan 2014, Liu et al. 2011). For example, Kluver and Konstan (2014) compared the performance of different algorithms for newly joined users with a limited number of ratings. They found that, when the number of ratings is very small (i.e., fewer than 10), non-personalized baseline algorithms performed better than did the more sophisticated CF-recommendation algorithms. As the number of ratings increases (i.e., more than 12), personalized algorithms, such as the well-known singular-value decomposition (SVD) algorithm, eventually outperform non-personalized algorithms. In addition, prior research has explored how the distribution of rating data may influence the performance of different recommendation algorithms at the dataset level. For example, Adomavicius and Zhang (2012) investigated how the characteristics of a dataset—rating space structure (measured by size, shape, and density), rating value (measured by the standard deviation of the rating), and rating frequency distribution (measured by Gini coefficients)—affect the performance of popular recommendation algorithms on the dataset. Prior literature has explored multiple variables related to data distribution. Table A1 in the Online Supplement provides a summary of these variables.

The vast majority of prior work that explored rating data characteristics, however, has focused primarily on the dataset-level rating characteristics and their influences on the aggregate performance of the recommendation algorithms for *all* users of the system (e.g., Adomavicius and Zhang 2012). The impacts of user-level rating characteristics have largely been overlooked in the literature. To the best of our knowledge, no prior work has provided a comprehensive investigation of user-level recommendation performance and the factors that influence the user-level performance. As discussed, user-level recommendation performance can directly shape a user's view of a recommender system and subsequently influence the user's acceptance of recommended items. In this research, we study this important issue through a structured and systematic exploration of the underlying relationships between various user-level rating characteristics (i.e., rating values, rating structure, and network embeddedness) and the predictive performance of popular recommendation algorithms at the individual-user level. Our research extends the extant literature by offering additional insights into which rating characteristics play a more (or less) important role and have positive (or negative) impacts on the performance of recommendation algorithms at the individual-user level.

2.2. Social Network Analysis in Recommender Systems

Extant literature shows that social influence in product consumption is important, because, according to social influence theory, one's beliefs, feelings, and behaviors are influenced by other people (Mason et al. 2007). Hence, data on social relations can be used to predict users' consumptions and preferences. As a result, a number of recent studies in recommender systems have started to integrate network analysis to model shared interests among users (e.g., Lee et al. 2015, Ting et al. 2016, Yang et al. 2012, Zhou et al. 2012). For example, Ting et al. (2016) explored different types of social data (e.g., liked music fans page, friends' liked music fans page, check-in locations) that can be collected from social networking websites and found that such social data can help in more accurate estimation of user preferences, thus improving performance of music recommender systems. Yang et al. (2012) proposed an approach that models social connections from social networking websites and uses them in recommendation algorithms. They showed that the new approach could outperform traditional algorithms, including matrix factorization and nearest neighbor methods. Further, Lee et al. (2015) developed a new method that uses historical networking data to predict user preferences and make recommendations.

Most of these prior studies have focused on *explicit* social networks, that is, networks constructed from explicit social relationships, such as Facebook friends. Many systems, including Amazon and Netflix, however, do not have explicit social relationships; therefore, it is not feasible to build explicit networks for such systems. Thus, in this research, we introduce a new approach to construct an *implicit* network based on users' co-rating patterns. Notably, our approach is not limited by the explicit social relations established among users and, therefore, can be applied to broader contexts.

3. Recommendation Algorithms and Performance Evaluation

3.1. Recommendation Algorithms

In a typical recommender system, we have a set of users U and a set of items I. The entire user-item space is denoted as $S = U \times I$. Each user rates a subset of all items with some numeric scores, e.g., on a scale from 1 to 5. Thus, the main recommendation problem is to predict the unknown ratings for items that have not yet been rated by a user. More formally, let R_{ui} represent the rating that user u gave to item i and D be the set of known ratings; then, the task of a recommendation algorithm is to estimate unknown R_{ui} values for all $(u, i) \in S \setminus D$ pairs. Once ratings for the yet-unrated items are estimated, the item(s) with the highest estimated rating(s) can be recommended to the user.

We seek to understand how popular standard algorithms perform for users with different rating characteristics. We focus on three CF recommendation algorithms that are widely used in many real recommender systems: *user-based nearest neighbor* approach, *item-based nearest neighbor* approach, and *matrix factorization* approach.

User-based and *item-based* CF approaches are centered on computing the relationships between users and between items, respectively. The user-based CF approach predicts unknown ratings of a user based on the ratings of the nearest neighbor users who have ratings similar to those of the target user (Adomavicius and Tuzhilin 2005, Konstan et al. 1997, Resnick et al. 1994), and an item-based CF approach predicts unknown ratings based on the ratings of the nearest neighbor items that receive similar ratings from users (Deshpande and Karypis 2004, Sarwar et al. 2001).

Taking the user-based CF approach for example, we compute the prediction of an unknown rating for a target user-item pair as the weighted sum of ratings on the item provided by the target user's neighbors (i.e., other users who have rating characteristics similar to those of the target user). More formally, the value of the unknown rating for user u on item i is an aggregate of item i's ratings provided by a set of users, $N(u,i)$, who have ratings patterns similar to that of user u and rated item i. The similarity between two users is used as a weight to aggregate ratings. The more similar user v and target user u are, the more weight that the rating provided by user v will carry. Thus, the predicted rating R^*_{ui} is computed as:

$$R^*_{ui} = b_{ui} + \frac{\sum_{v \in N(u,i)} sim_{uv} \times (R_{vi} - b_{vi})}{\sum_{v \in N(u,i)} |sim_{uv}|} \tag{1}$$

We preprocess the rating data by removing baseline effects from known ratings (Bell and Koren 2007). The *baseline* estimate for each user-item (u,i) pair, denoted as b_{ui}, is computed as $b_{ui} = \mu + b_u + b_i$, where μ is the global mean of ratings in the dataset and b_u (b_i) is the average rating deviation of user u (item i) from global mean μ. We compute the similarity between users sim_{uv} using the vector cosine similarity metric.

The *matrix factorization* technique is a model-based approach that characterizes both items and users by latent factors inferred from the rating matrix. It maps both users and items to a joint latent factor space of dimensionality k (Koren et al. 2009). Predictions are made by taking the inner product of the user and item vectors. Our experiments used the well-known canonical version of matrix factorization, also referred to as "FunkSVD" (Funk 2006, Koren et al. 2009). In the FunkSVD model, each user u is modeled by a latent vector $p_u \in \mathbb{R}^k$ (the user's preferences for K features), and each item i is modeled by a latent vector $q_i \in \mathbb{R}^k$ (the extent to which the item possesses the corresponding K features). FunkSVD uses the inner product of the user and item vectors to estimate how much a user u likes an item i, denoted by R^*_{ui}, beyond what already can be captured by the baseline estimate b_{ui}. Hence, the preference rating is predicted as:

$$R^*_{ui} = b_{ui} + p_u^T q_i \tag{2}$$

The user and item latent vectors p_u and q_i are estimated by a stochastic gradient descent learning technique. The learner loops through each known rating to estimate the associated prediction error $e_{ui} = R_{ui} - R^*_{ui}$ and updates the values of user and item latent factors. Given a learning rate parameter γ and regularization parameter λ, the p_u and q_i are iteratively updated as follows:

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i) \qquad \textit{and} \qquad p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$

The learning iteration continues until the minimum improvement in predictive accuracy or a predefined number of iterations is reached.

3.2. Performance Metric

The prior literature characterizes recommender systems performance as a multi-faceted construct that can be evaluated along a number of dimensions (see Herlocker et al. 2004, Shani and Gunawardana 2011, for an overview). Among various performance measures for evaluating recommender systems, recommendation accuracy has always been a major focus. In this work, we focus on the popular *predictive accuracy* metric in the recommender systems literature (Herlocker et al. 2004, Shani and Gunawardana 2011).

Predictive accuracy is a measure of the ability of a recommender system to predict users' preference ratings for items correctly (i.e., how well a system can identify relevant vs. irrelevant items for each user). Predictive accuracy is calculated by comparing the predicted ratings estimated by a recommendation algorithm against the actual ratings submitted by the users. In our study, recommendation accuracy is measured using the *weighted root mean squared error* (*wRMSE*) (Hu et al. 2008). We also have explored other well-known accuracy metrics, such as *root mean squared error (RMSE)* and *mean absolute error (MAE)*. In our analysis, we find all of these accuracy metrics to be highly correlated with each other; therefore, the results are consistent across these accuracy metrics. In the remainder of this paper, we will focus on the results for wRMSE. We report the results for RMSE and MAE in the Online Supplement. For a given user u, the $wRMSE_u$ is defined as:

$$wRMSE_u = \sqrt{\sum_{(u,i)\in T_u} w_{ui}(R^*_{ui} - R_{ui})^2 / \sum_{(u,i)\in T_u} w_{ui}} \tag{3}$$

where T_u is the set of ratings by user u used for performance evaluation, R^*_{ui} represents the system-predicted rating for user u on item i, R_{ui} is the actual rating provided by user u for item i, and w_{ui} is weight calculated based on predicted rating value R^*_{ui}.

4. Rating characteristics

We now discuss the choice of variables for rating characteristics measured for the individual users who are examined in this study. In statistical models, the choice of independent variables is either theory-driven (e.g., confirmatory research), or exploration-driven (e.g., exploratory research). In the first category, the potential impact of different variables can be hypothesized based on existing references theories, which can help to eliminate alternatives to focus on a small set of variables. Our study falls into the second category, which is used in situations in which there is an insufficient theoretical foundation. In our case, there is a significant lack of understanding in the literature about what data characteristics affect user-level performance of recommendation algorithms and why. Only recently has some research started to examine how data influence the recommendation algorithms at the aggregate dataset level (Adomavicius and Zhang 2012, Huang and Zeng 2011). Due to the insufficient theoretical foundation, in this work, we use an exploration-driven approach based on the specialized structure of the recommendation data to select explanatory variables.

The structure of the recommendation data can be viewed as a partially filled rating matrix with each row as representing a user and each column as representing an item. Each cell represents a rating given to an item by a user, and only some subset of cells has known values (i.e., known ratings submitted to the system). Each user's rating data are represented by a one-dimensional, partially-filled rating vector. Thus, users' rating characteristics can be viewed as the characteristics of a sparse vector. We explore three fundamental types of rating vector characteristics: *rating value*, *rating structure*, and *neighborhood network embeddedness*. Rating value refers to properties of the actual numeric values of known ratings; rating structure is the density and distribution of consumed items; and neighborhood network embeddedness concerns the connection and overlap of the focal user's rating vector with other people in the user space.

The first two categories—rating value and rating structure—are calculated based solely on each focal user's rating records, meaning that other users' data are not considered for constructing these measurements. In contrast, the third category introduces the network structure measures that are calculated based on the entire user space (including both the focal user and others who use the same system). The network measurements represent the embeddedness of a focal user in a network constructed from the rating dataset.

Note that, within each of the three categories, there are many possible variables that are representative of the nuances of users' rating data. For each of the three categories that we explored, we started with a large pool of variables that we could find in the literature, and eliminated the ones that are highly correlated with other relevant variables or that did not add explanatory power in the regression analysis. The resulting set of variables in each of the three categories is presented in the following subsections. Table 2 provides an overview of the user-level rating characteristics that we include in the regression model of this study.

Table 2 Measures for User-Level Rating Characteristics.

Category	Measurement	Description
Rating value	Rating mean	The average rating value of all ratings made by a user.
	Rating variance	The variance of ratings made by a user.
Rating structure	Density	The percentage of items rated by a user over the total number of items.
	Popularity mean	The average popularity score of items rated by a user.
Neighborhood network embeddedness	Eigenvector centrality	The extent to which a user is similar to his or her neighbors in a weighted network.
	Clustering coefficient	The level of cohesion of a user's local network.

4.1. Rating Value

We use rating mean and rating variance to characterize users' rating values.

Rating Mean. Rating mean is calculated as the average rating value provided by a user and represents the central tendency of a user's rating-value distribution. It captures the user-specific bias in providing preference ratings, i.e., some users may systematically tend to give higher ratings than do others, as represented by higher rating mean, and some users may tend to be more critical when rating items, as represented by a lower rating mean. Thus, many recommendation techniques involve a preprocessing step that normalizes rating data by subtracting the user-rating mean. This normalization step not only can improve the prediction accuracy significantly, but also can reduce the possible influence of a user's rating mean on the performance of recommendation techniques.

Rating Variance. Rating variance is a measure of the variations of all the ratings for a target user. Prior literature has suggested that, at the population level, high rating variance can be a potential cause of recommendation errors (Herlocker et al. 2000). Rating variance is reported to have a linear and negative impact on the overall recommendation accuracy over the entire user population (Adomavicius and Zhang 2012). In this research, we conjecture that an individual user's rating variance has a negative impact on the prediction accuracy of recommendation algorithms for the user.

4.2. Rating Structure

We capture users' rating structure by rating density and average popularity of consumed items.

Density. Density is a structural measure that captures the percentage of items that a user has rated, i.e., $Density = |I_u|/|I|$, where I_u is the set of items that user u has rated and I is the set of all items. The operating assumption in the recommender systems literature is that rating density can have a positive influence on the performance of recommendation algorithms (e.g., Chee et al. 2001, Resnick et al. 1994, Sarwar et al. 1998). That is, dense rating data would lead to higher accuracy than would sparse rating data. With higher density (i.e., more ratings provided), recommendation algorithms have more information to model users' preferences and to generate more accurate predictions. We will explore this relationship in our study for users with different rating density levels. In our analysis, we applied logarithm transformation to the density variable to normalize its distribution.

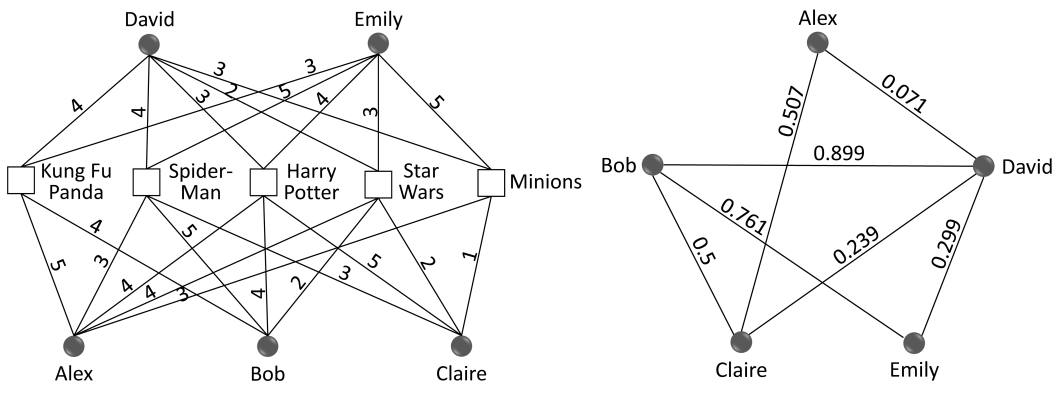
Popularity Mean. We use the popularity distribution of items rated by the user to measure the structure of users' rating vector. Intuitively, items with more ratings are considered to be more popular; hence, we calculate an item's popularity as the number of users who have rated the item (i.e., total number of ratings received by the item). We then normalize the popularity score by dividing it over the total number of possible ratings (i.e., total number of users). That is, for a given item i, we define $popularity_i = |R_i|/|U|$, where R_i is the set of ratings received by item i, and U is the set of all users. To calculate the average popularity of a user u's rating vector, we extract all of the items rated by u, denoted as I_u, and calculate the mean popularity of these items as:

$$PopularityMean_u = \frac{\sum_{i \in I_u} popularity_i}{|I_u|} = \frac{\sum_{i \in I_u} |R_i|/|U|}{|I_u|} \tag{4}$$

Popularity mean captures a user's tendency to rate popular vs. unpopular items. A higher popularity mean indicates that the user has rated more popular items in the past. When other variables are held equal, this means that the user is likely to share more commonly rated items with other users in the system, making it easier for many recommendation algorithms, such as the user-based CF technique, to identify nearest neighbors who have tastes that are similar to those of the target user. This, in turn, helps the algorithms to make more accurate predictions.

4.3. Neighborhood Network Embeddedness

In this research, we build a neighborhood network based on users' co-rating relations. Recommender systems typically include a list of users and a list of items. Therefore, it is intuitive to construct a two-mode network to describe rating data (Wasserman and Faust 1994). In the two-mode network, users and items are represented by two different types of nodes, and ratings are represented by connections between users and items. It is worth noting that the two-mode network does not allow connections between same types of nodes. We can, however, convert the two-mode network into a one-mode network, either user based or item based, to describe the relationships among users or among items. In this study, we focus on the user-based, one-mode neighborhood network in which two users are connected if they have similar rating characteristics. We use a movie-rating example (Figure 1a) to demonstrate our network construction process.



(a) Two-mode Affiliation Network (b) One-mode Neighborhood Network

Figure 1 Two-Mode and One-Mode Networks for a Movie Rating Example

Figure 1a provides a two-mode network visualization of a movie rating. Items are represented by squares, and users by circles. Connections between nodes indicate users' ratings of items. For instance, Alex rated "Kung Fu Panda" 5 stars; thus, there is a line that connects Alex to "Kung Fu Panda" with a value of 5. The two-mode network can then be converted to a user-based, one-mode neighborhood network, illustrated in Figure 1b, based on users' co-rating relations.

In a user-based neighborhood network (e.g., Figure 1b), all nodes represent users. Two users are connected if they have similar rating characteristics (i.e., considered to be "neighbors"). We use a Pearson correlation coefficient as the similarity measure and consider two users to be neighbors if they meet two conditions: (1) the two users must have rated at least three items in common; and (2) the Pearson similarity coefficient between the two users' ratings is positive. The first condition is imposed due to a mathematical constraint that requires at least three values to calculate Pearson similarity between two vectors. The second condition is included because only positive values of the Pearson measure indicate that rating characteristics are similar. Negative values of the Pearson measure indicate that ratings are dissimilar and that those users are not considered neighbors in the network (i.e., no connections are established). The connections in the neighborhood network are weighted based on the similarity between two users' ratings on common items.

In the example shown in Figure 1b, user Alex is connected to two other users, Claire and David. The value of each connection indicates the extent to which Alex's ratings are similar to those of his neighbors. In this case, Alex has higher similarity in movie preferences to Claire (0.507) than to David (0.071). The fact that there is no connection between Alex and Emily, despite their having rated five movies in common, is because their movie preferences are quite different, as reflected by a negative Pearson similarity score (-0.896). Although users' own past ratings are important inputs for recommendation algorithms to generate high-quality recommendations, prior research also suggests that additional insights can be derived from the network of relations (Huang and Zeng 2011, Ransbotham et al. 2012). A common way to assess the value is through network embeddedness, which is a measure of the extent to which a user is connected to other users in the network (Grewal et al. 2006, Kadushin 2012, Yan et al. 2015). In our user-based neighborhood network, each user's network embeddedness represents the number of social resources that can be leveraged to model the user's preferences and make accurate predictions. Prior literature has developed a number of measures for network embeddedness, including degree, eigenvector, closeness, betweenness, and clustering coefficient (Wasserman and Faust 1994). In this research, we use eigenvector centrality and clustering coefficient to calculate the network embeddedness for each user.

Eigenvector Centrality. Centrality is a measure of the number and strength of direct connections possessed by a user (Wasserman and Faust 1994). There are many different ways to measure types of centrality, including degree centrality, betweenness centrality, and eigenvector centrality. We use the eigenvector centrality metric in the model. The eigenvector centrality of a user is calculated based on the weighted sum of the centralities of the user's neighbors. Users in the network who are more closely connected to other well-connected users, i.e., those who already occupy important positions in the network, will have higher eigenvector centrality values. For instance, in Figure 1b, both Bob and Claire have three connections. However, given that Bob and Claire are connected to different users and different similarity values are associated with their connections, the eigenvector centrality measures of the two users are different: Bob's eigenvector centrality is 0.639, which is larger than that of Claire (0.341). The reason is that two out of three neighbors of Alex are closely connected to him through high similarity scores (i.e., 0.899 with David and 0.761 with Emily). In contrast, despite the fact that Claire also has three neighbors, her connections to her neighbors Alex and David do not carry high values (i.e., 0.507 and 0.239, respectively).

Using eigenvector centrality ensures that we not only consider the number of connections of a focal user, but also the weights of these connections in predicting his or her item preferences. The operating assumption in CF recommendation algorithms is that users who have similar preferences for consumed items also tend to have similar preferences on unconsumed items. Therefore, with other variables' being equal, users who have higher eigenvector centrality values tend to share more similar tastes with others in the network. Thus, it is easier for recommendation algorithms to build an individual user's preference model, which helps to generate more accurate recommendations for the target user.

Clustering Coefficient. The clustering coefficient is a measure of the transitivity of one's local network. It captures the dense relations among a subgroup of users in the network (Wasserman and Faust 1994). By focusing on triangles, the clustering coefficient measures the extent to which the neighbors of a user are also connected to each other. Prior literature has found that the clustering coefficient helps to explain why certain CF algorithms perform well for transaction-based recommendation datasets (Huang and Zeng 2011). For a given user u, the clustering coefficient can be calculated as follows:

$$ClusteringCoefficient_u = \frac{2|E(G_u)|}{deg_u \times (deg_u - 1)} \tag{5}$$

where deg_u denotes number of connections that user u has in the network, G_u is the set of neighbors of user u, and $E(G_u)$ is the set of connections among user u's neighbors. Take Alex in Figure 1b as an example. Alex has two neighbors, Claire and David, i.e., $deg_{Alex} = 2$. Because Claire and David are connected with each other, there is one connection between them, i.e., $E(G_{Alex}) = 1$. Therefore, the clustering coefficient for Alex is $2 \times 1/(2 \times (2-1)) = 1$. Likewise, the clustering coefficient for Bob is 0.667. Alex has a larger clustering coefficient than does Bob ($1 > 0.667$), which suggests that Alex is in a more cohesive neighborhood network than is Bob. The high cohesion of a neighborhood network allows the user-based CF algorithm to find similar users and to utilize their information to generate more accurate predictions for the target user.

5. Experiments

We first explain our datasets and then describe how our experiments were conducted.

5.1. Datasets

We used five real-world rating datasets in our experiments to examine the relationship between users' rating characteristics and user-level predictive performance of popular recommendation algorithms. Our datasets come from different sources and have different characteristics, including the size of the rating space (i.e., number of users and number of items represented), the number of ratings available, and the rating density (i.e., proportion of the rating space that is known). The five datasets include: (1) MovieLens 100K movie rating dataset (Harper and Konstan 2016); (2) a subset from the MovieLens 1M dataset (Harper and Konstan 2016); (3) a subset from Netflix 100M rating dataset (Bennett et al. 2007); (4) a subset of Yahoo! Music rating data (from `http://webscope.sandbox.yahoo.com/`); and (5) a subset of Yelp restaurant rating data (from `https://www.yelp.com/dataset_challenge`).

The MovieLens 100K movie rating dataset contains 943 users and 1,682 movies. Because it is a small dataset, and all users have at least 20 ratings, we kept all observations in this dataset. For the MovieLens 1M and Yelp datasets, to ensure that there are enough ratings for recommendation algorithms to analyze, we filtered these two datasets to include users with at least 20 ratings and items with at least 20 ratings (which is a common practice in recommender systems literature; see Herlocker et al. 2004). The resulting subset of MovieLens 1M includes 6,022 users and 3,043 movies, and the subset of Yelp includes 5,548 users and 4,849 restaurants. Because the datasets of Netflix and Yahoo! Music contain a large number of users and items, we first extracted a random sample of approximately 5,000 users from each dataset and then filtered the samples to include users and items that have at least 20 ratings. Our resulting Netflix subset includes 4,998 users and 5,725 movies, and our Yahoo! Music subset includes 4,930 users and 21,649 songs. All ratings in the five datasets are integer values between 1 and 5, where 1 represents the least-liked items and 5 represents the most-liked items. Table B1 in the Online Supplement presents a summary of the five datasets.

5.2. Experimental Procedure

Our experiments focused on the predictive accuracy of three recommendation techniques that are widely used in real-world recommender systems: user-based and item-based CF algorithms and the matrix factorization FunkSVD algorithm. We used LensKit, an open-source recommendation toolkit, to run the three recommendation algorithms (Ekstrand et al. 2011). Before applying any algorithm, we applied a preprocessing step that normalized ratings by subtracting the user-item baseline score (Bell and Koren 2007).

To tune the algorithm parameters for item-based and user-based CF algorithms, we first ran the algorithms with both the cosine similarity and Pearson correlation coefficient as similarity metrics. Our results showed that cosine similarity slightly outperformed Pearson correlation on all of the datasets. Hence, we chose to use cosine similarity in the final experiments. We also experimented with different neighborhood sizes, ranging from 5 to 200. In general, accuracy increased as we increased the number of neighbors. After about 30 neighbors, however, the improvement gains diminished, and the accuracy stabilized. In our experiments we used 50 as the final neighborhood size. For FunkSVD, we varied the number of latent vectors from 1 to 200. We found that, after about 20 features, the improvement gain of accuracy diminished. We used 50 as the number of latent factors for FunkSVD in the experiments.

We applied the standard holdout validation method to estimate the predictive accuracy (measured by *wRMSE*) of the three recommendation algorithms for each user. Specifically, we randomly divided the rating data into two subsets: 80% for training and 20% for testing. Our data split operated at the user level to ensure that each training sample contained 80% of random ratings for each user and that the corresponding test sample contained the remaining 20% of ratings for each user. We applied the three recommendation algorithms on the training sample (80%) to build a model of users' preferences. The *wRMSE* of each algorithm was calculated for each user based on the holdout observations in the test sample (20%).

We repeated the holdout validation process 30 times, each time with a different random split of training and testing samples. We then computed the average *wRMSE* for each user and each algorithm across 30 runs. We used the mean *wRMSE* as the final predictive accuracy, which was later used as the dependent variable in the regression analysis. Using the mean *wRMSE* values across multiple runs helped us to mitigate possible noise and bias that result from random data splits.

Our next step was to calculate each user's rating characteristics and to use these characteristics as independent variables in the regression analysis. As discussed in Section 4, we focus on three categories of rating characteristics: rating value, rating structure, and neighborhood network embeddedness. Our final analysis included six independent variables from these three categories, *RatingMean*, *RatingVariance*, *Density*, *PopularityMean*, *EigenvectorCentrality*, and *ClusteringCoefficient*, and three dependent variables, mean *wRMSE* calculated for item-based CF, user-based CF, and FunkSVD. Table B2 in the Online Supplement provides the summary statistics for all of the variables.

6. Analysis and Results

Using the aforementioned six measures of users' rating characteristics as independent variables, we built a regression model to explain the predictive accuracy (as measured by *wRMSE*) for each user. We conducted the analysis separately for each recommendation algorithm. Within each recommendation algorithm, we expect that the effects of rating characteristics would be largely consistent across datasets. Across the three algorithms, because the three techniques use different ways to compute predictions for unknown ratings, we expected that certain rating characteristics would have different influences on the accuracy performance of different algorithms. We first performed regression analysis within each dataset and then pooled all five datasets together for an overall cross-dataset analysis.

6.1. Within-Dataset analysis

We started by analyzing the data separately for each dataset. We mean-centered all independent variables before conducting any analysis. We then tested nonlinear forms of independent variables and found the quadratic term *RatingVariance* to be significant. We also conducted a thorough examination to check whether our data violated any of the least squares regression assumptions. Our examination did not raise any multicollinearity issues (all variance inflation factor [VIF] values were below 10, and the average VIF values ranged from 1.59 to 4.02). We also did not have normality or outlier issues. Therefore, we used ordinary least square (OLS) analysis with robust standard error estimation through minimizing the sum of squared distance between actual and fitted values. Our regression model is presented below:

$$\begin{aligned} wRMSE_u = {} & \beta_0 + \beta_1 \times RatingMean_u + \beta_2 \times RatingVariance_u + \beta_3 \times RatingVariance_u^2 \\ & + \beta_4 \times lnDensity_u + \beta_5 \times PopularityMean_u + \beta_6 \times EigenvectorCentrality_u \\ & + \beta_7 \times ClusteringCoefficient_u + \epsilon_u \end{aligned} \quad (6)$$

The regression results for the three recommendation algorithms for the five datasets are summarized in Table 3. Overall, our regression model provides a good fit for all five datasets. The overall average R^2 value is 83.67%, suggesting that the six independent variables used in our model are able to explain a large amount of variations in the dependent variable (*wRMSE*) for all three algorithms and five datasets. Among the five datasets, the model provides the best fit for the Yahoo! Music data, with R^2 values of 90.77%, 92.39%, and 89.65% for item-based CF, user-based CF and FunkSVD, respectively. That is, approximately 90% of the variance in algorithms' predictive accuracy for users of Yahoo! Music could be explained by only six variables of users' rating characteristics. Similarly, the six variables could explain 82.63%, 85.25%, and 82.59% of the variations in *wRMSE* for the MovieLens 100K dataset; 78.64%, 82.97% and 75.22% for the MovieLens 1M dataset; 85.83%, 88.96% and 82.99% for the Netflix dataset; and 78.65%, 78.44%, and 80.13% for the Yelp dataset. Taken together, the results of the analysis suggest that the user-based CF has the best model fit on four out of five total datasets.

Turning to individual variables, we find that the estimated coefficients are highly consistent across datasets and algorithms. First, both variables of rating value (i.e., *RatingMean* and *RatingVariance*) have consistent effects on the dependent variable (*wRMSE*) across datasets and algorithms.

Table 3 Summary of Regression Results within Each Dataset.

Dataset	DV: wRMSE	Item-based CF	User-based CF	FunkSVD
MovieLens 100K	RatingMean	-0.0217**	-0.0229**	-0.0310***
	RatingVariance	0.4240***	0.4312***	0.4230***
	RatingVariance^2	-0.1044***	-0.0747***	-0.0760***
	lnDensity	-0.0650***	-0.0650***	-0.0539***
	PopularityMean	-0.3167***	-0.3313***	-0.1813†
	EigenvectorCentrality	-7.9472***	-8.4880***	-8.4831***
	ClusteringCoefficient	-0.6515***	-0.7697***	-0.6862***
	(Constant)	0.9351***	0.9387***	0.9337***
	R^2	0.8263	0.8525	0.8259
	No. of observations	943		
MovieLens 1M	RatingMean	-0.0413***	-0.0429***	-0.0408***
	RatingVariance	0.3916***	0.4085***	0.3993***
	RatingVariance^2	-0.0838***	-0.0619***	-0.0841***
	lnDensity	-0.0519***	-0.0592***	-0.0398***
	PopularityMean	-0.1572**	-0.1835***	-0.1567**
	EigenvectorCentrality	-20.3892***	-22.1144***	-19.8714***
	ClusteringCoefficient	-0.6423***	-0.8562***	-0.6183***
	(Constant)	0.8730***	0.8847***	0.8882***
	R^2	0.7864	0.8297	0.7522
	No. of observations	6022		
Netflix	RatingMean	-0.0186***	-0.0162***	-0.0195***
	RatingVariance	0.4380***	0.4525***	0.4472***
	RatingVariance^2	-0.0941***	-0.0781***	-0.0979***
	lnDensity	-0.0568***	-0.0553***	-0.0492***
	PopularityMean	-0.2143***	-0.2102***	-0.2624***
	EigenvectorCentrality	-13.8838***	-13.9624***	-14.0587***
	ClusteringCoefficient	-0.7916***	-0.8408***	-0.8322***
	(Constant)	0.8952***	0.9070***	0.9123***
	R^2	0.8583	0.8896	0.8299
	No. of observations	4998		
Yahoo! Music	RatingMean	-0.0220***	-0.0234***	-0.0481***
	RatingVariance	0.3626***	0.3721***	0.3784***
	RatingVariance^2	-0.0565***	-0.0530***	-0.0587***
	lnDensity	-0.0648***	-0.0571***	-0.0438***
	PopularityMean	-0.6764***	-0.7364***	-0.7553**
	EigenvectorCentrality	-5.1996***	-4.7571***	-6.4931***
	ClusteringCoefficient	-0.5806***	-0.5405***	-0.6464***
	(Constant)	1.2233***	1.2293***	1.2737***
	R^2	0.9077	0.9239	0.8965
	No. of observations	4930		
Yelp	RatingMean	-0.0324***	-0.0286***	-0.0323***
	RatingVariance	0.4853***	0.4552***	0.4662***
	RatingVariance^2	-0.0524***	-0.0468***	-0.0499***
	lnDensity	-0.0571***	-0.0377***	-0.0202**
	PopularityMean	-0.9461*	-1.1757**	0.1659
	EigenvectorCentrality	-3.1582***	-3.0339***	-3.5022***
	ClusteringCoefficient	-0.2244***	-0.1789***	-0.1768***
	(Constant)	0.9896***	0.9806***	0.9603***
	R^2	0.7865	0.7844	0.8013
	No. of observations	5548		

Note: $^{***}p \leq .001, ^{**}p \leq .01, ^{*}p \leq .05, \dagger p \leq 0.1$. We mean-centered all independent variables for the analysis.

The effect of *RatingMean* is negative on *wRMSE* and significant in all 15 cases. The results are different from those of our earlier conjecture that *RatingMean* should have no significant impact on prediction accuracy. Recall that we normalized the data by removing the baseline effect for each user-item pair, which is a combination of the global mean, user, and item mean deviations, from all ratings (Bell and Koren 2007). This step is expected to reduce possible biases introduced by users' mean ratings (e.g., systematically high or low mean ratings) on the algorithm's rating prediction process. Our analysis results, however, show that, even after normalizing rating data, *RatingMean* could still play a significant role in the predictive accuracy of recommendation algorithms. The negative and significant coefficients of *RatingMean* on *wRMSE* indicate that *RatingMean* is positively related to the accuracy performance of all three recommendation algorithms.

RatingVariance has a positive and significant effect on *wRMSE* across all datasets and algorithms (i.e., all 15 coefficients are significant at the 0.001 level). This finding is consistent with our expectation that more variance in ratings implies a higher level of uncertainty in a user's preferences, making it more difficult for recommendation algorithms to make accurate estimations for the user. In addition, the squared term $RatingVariance^2$ is also significant for all datasets and algorithms, indicating a non-linear effect of *RatingVariance* on *wRMSE*. Specifically, the combination of a large positive linear coefficient (ranging between 0.3626 and 0.4853) and a small negative quadratic coefficient (ranging between -0.0468 and -0.1044) suggests that the relationship between *RatingVariance* and *wRMSE* follows a concave shape that monotonically increases within the range of possible values. Figure 2 provides an illustration of such a relationship in the Yahoo! Music dataset. As shown in the figure, as the value of *RatingVariance* increases, the *wRMSE* of the recommendation algorithms monotonically increases (due to the positive overall effect). The effect of *RatingVariance*, however, weakens as its values increase (due to the negative quadratic effect). In other words, when rating variance is small, a unit of change in *RatingVariance* has a larger impact on *wRMSE* than when rating variance is large.

For individual users, there are multiple possible causes for rating variance. First, rating variance may be caused by inconsistency in users' preferences on items. Prior literature has reported that users provide inconsistent ratings even when they are asked to rerate the same items (Cosley et al. 2003). Second, rating variance may result from the range of items that users choose to rate. For example, two users with the same movie preferences might choose to rate different movies in the system and, therefore, have different rating variance. The presence of users who rate a broad range of items might result in larger rating variance than would result from those who concentrate only

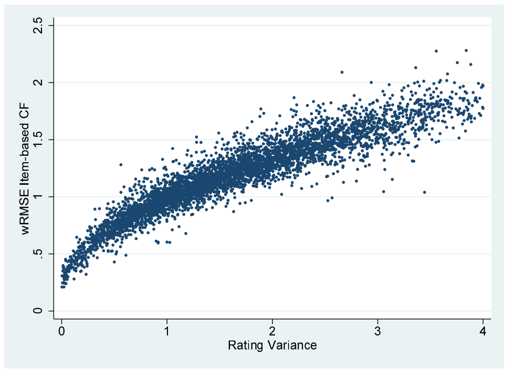


Figure 2 Plot of RatingVariance and wRMSE of Item-Based Collaborative Filtering on Yahoo! Music Dataset.

on items that they like. Third, users might apply the rating scale differently. Some users tend to rate items with extreme scores (e.g., 1 for disliked items and 5 for liked items), whereas others tend to use moderate scores for their ratings (e.g., use 3 for disliked items and 4 for liked items). Large rating variance could make it more difficult for recommendation algorithms to extrapolate users' true preferences, leading to higher prediction errors.

With respect to users' rating structure, both variables, *lnDensity* and *PopularityMean*, are significant for most datasets and algorithms. Coefficients of consumption density (*lnDensity*) are negative and statistically significant in all 15 cases. The results indicate that the density of the rating seems to play a significant role in all models. The negative coefficient of *lnDensity* shows that, with other independent variables' being equal, the more ratings that a user provides (i.e., higher consumption density), the more accurate the predicted ratings for the unknown items become for all three algorithms. In other words, recommendation algorithms perform better with more ratings from a user. As we expected, as a user submits more ratings, more information is available to be utilized by recommendation algorithms to better model users' preferences (e.g., by finding true neighboring users/items or by accurately extracting user and item latent features), and to make more accurate predictions. Thus, we observe that users with denser rating vectors tend to receive more accurate predictions.

PopularityMean also shows a significant effect across datasets and algorithms (only one out of 15 coefficients are insignificant). The consistent negative sign of all coefficients suggests a positive relationship between *PopularityMean* and prediction accuracy for all three algorithms. This is consistent with our earlier conjecture that recommendation algorithms make more accurate predictions for users who have rated more popular items. The consumption of popular items allows users to have more items in common with other users, which helps CF recommendation algorithms to better model users' preferences (e.g., by correctly identifying neighbors with similar rating characteristics) and, therefore, to make more accurate rating predictions. In comparison, for users who consume more niche items, it is more difficult for algorithms to model their preferences due to the lack of related information (e.g., there may not be many neighboring users or items with similar rating characteristics); hence, the rating predictions could be less accurate.

Finally, for neighborhood network measures, both *EigenvectorCentrality* and *ClusteringCoefficient* show consistently significant effects on *wRMSE* across all datasets and algorithms, indicating that neighborhood network variables have a strong influence on the prediction accuracy of recommendation techniques. For *EigenvectorCentrality*, we have negative and significant coefficients across all datasets and algorithms (all 15 coefficients are significant at the 0.001 level), providing strong support for our prior conjecture. That is, our results show a strong positive influence of *EigenvectorCentrality* on predictive accuracy. In other words, recommendation algorithms make more accurate predictions for users who are connected with a large number of neighbors who share highly similar preferences (i.e., higher weights of connections). High centrality values, therefore, are beneficial for the recommendation algorithms to identify true neighbors who have the most similar rating characteristics, resulting in better predictive accuracy for the target user.

Likewise, we have negative and significant estimated coefficients for *ClusteringCoefficient* at the 0.001 level in all 15 cases. This means that users who are embedded in a more cohesive local network tend to receive more accurate prediction ratings. This is because a high value of *ClusteringCoefficient* means that the target user's local network is more clustered and his or her neighbors share more common items. Thus, the high cohesion of local network further enables CF algorithms to utilize many similar users' rating information for making predictions for the target user. Overall, the two network variables *EigenvectorCentrality* and *ClusteringCoefficient*, each of which captures different aspects of network embeddedness, are among the strongest indicators of predictive accuracy of recommendation algorithms.

We further estimated the relative importance of individual variables on the predictive accuracy of recommendation algorithms by examining the marginal contributions in R^2. Table B3 in the Online Supplement presents our results. First, for each variable, we performed an F-test to compare the difference between the full model and the model without the variable. All of the F-test results were significant at either the 0.001 or 0.01 level, indicating the significant contributions of all six variables to the fit of the full model. Second, by comparing the relative importance of the six variables, we find that *RatingVariance* is consistently the most important variable in predicting accuracy for all algorithms. The next most important variables are *EigenvectorCentrality* and *Density*. This is followed by *ClusteringCoefficient* and *PopularityMean*. The least influential factor in most cases is

RatingMean. Third, by comparing the importance of each variable across three recommendation algorithms, we find that user-based CF is more sensitive to *RatingVariance* than are both item-based CF and FunkSVD. We also find FunkSVD to be less sensitive to *Density* but generally more sensitive to *EigenvectorCentrality* than are the user-based and item-based CF algorithms. Such comparisons provide insight for recommender system developers for customizing the system for individual users, based on their rating characteristics.

In summary, our within-dataset regression analyses demonstrated that a large amount of variation in the predictive accuracy of recommendation algorithms can be explained by using only six variables that describe users' rating characteristics. The effects of all six predictors are mostly significant and consistent across different datasets and application domains. The effects also seem to be universal for three popular CF recommendation algorithms (i.e., item-based CF, user-based CF, and FunkSVD).

6.2. Across-Dataset Analysis

We next combined all five datasets to analyze the effects of rating characteristics on predictive accuracy in one general framework. The overall across-dataset analysis helps us to explore possible effects of different datasets and application domains. Table B4 in the Online Supplement provides a summary of our combined dataset, and Table B5 presents the descriptive statistics for all variables.

We performed a regression analysis on the combined dataset, using regression with robust errors and a fixed effect of dataset. *Dataset* is introduced as a category variable with five values (i.e., MovieLens 100K, MovieLens 1M, Netflix, Yahoo! Music, and Yelp) to the model. We coded the MovieLens 100K as the baseline dataset in the analysis and compared the other datasets to the baseline. The regression model for the combined dataset is given below:

$$\begin{aligned} wRMSE_{du} = {} & \beta_0 + \beta_1 \times RatingMean_{du} + \beta_2 \times RatingVariance_{du} + \beta_3 \times RatingVariance^2_{du} \\ & + \beta_4 \times lnDensity_{du} + \beta_5 \times PopularityMean_{du} + \beta_6 \times EigenvectorCentrality_{du} \\ & + \beta_7 \times ClusteringCoefficient_{du} + Dataset_d + \epsilon_{du} \end{aligned} \tag{7}$$

Table 4 presents our regression results for the combined dataset. We report the coefficients in their original forms. The results of the overall analysis are highly consistent with the within-dataset results reported in Section 6.1. For the combined dataset, R^2 values for item-based CF, user-based CF, and FunkSVD are 85.35%, 87.2%, and 85.88%, respectively. Using six independent variables of users' rating characteristics, our models were able to explain, on average, 86.14% of the variations

Table 4 Regression Results for Combined Dataset.

DV: wRMSE	Item-based CF	User-based CF	FunkSVD
RatingMean	-0.0234***	-0.0234***	-0.0376***
RatingVariance	0.4011***	0.4059***	0.4075***
RatingVariance^2	-0.0620***	-0.0552***	-0.0633***
lnDensity	-0.0509***	-0.0486***	-0.0390***
PopularityMean	-0.2317***	-0.2220***	-0.2078***
EigenvectorCentrality	-7.3276***	-7.5772***	-7.3864***
ClusteringCoefficient	-0.4263***	-0.4744***	-0.4896***
(Constant)	0.9273***	0.9351***	0.9314***
MovieLens 1M	-0.0582***	-0.0515***	-0.0469***
Netflix	-0.0390***	-0.0329***	-0.0265***
Yahoo! Music	0.3000***	0.2958***	0.3457***
Yelp	0.0645***	0.0474***	0.0320***
R^2	0.8535	0.8720	0.8588
No. of observations	22,441		

Note: $^{***}p \leq 0.001$. We mean-centered all independent variables for the analysis.

in predictive accuracy of item-based CF, user-based CF, and FunkSVD algorithms when applied to 22,441 users from five different application domains.

All six independent variables have significant and consistent effects on *wRMSE* for the three recommendation algorithms in the overall analysis. *RatingMean*, *lnDensity*, *PopularityMean*, *EigenvectorCentrality*, and *ClusteringCoefficient* have negative and significant coefficients, indicating the positive impacts of these rating characteristics on recommendation algorithms' predictive accuracy performance. *RatingVariance* has a positive coefficient on *wRMSE*, demonstrating a strong negative impact of rating variance on the prediction accuracy of recommendation algorithms. The negative coefficient of the quadratic term $RatingVariance^2$ further suggests that the negative impact of rating variance on accuracy is nonlinear and decreases as the value of rating variance increases.

In summary, our pooled regression results on the combined dataset are highly consistent with the results of within-dataset analysis. After combining all datasets, the effects of all independent variables on *wRMSE* remain consistent and significant for all three algorithms. The fact that we still get consistent results after controlling the fixed dataset-specific effects suggests that the effects of the six variables explored in this study are not dataset-specific; instead, their effects are applicable to a variety of application domains for all three popular recommendation algorithms.

6.3. Robustness Check

We conducted a number of experiments to examine the robustness of our findings. First, we conducted a time-series analysis based on the timestamp information associated with user ratings. This helps us to check whether the findings are sensitive to possible changes in users' rating characteristics over time. All but the Yahoo! Music datasets include a timestamp for each rating. We split the rating data based on the timestamp, such that the training set contained 80% of ratings that users submitted in the earlier time frame, and the evaluation set contained the remaining 20% of ratings that were submitted during a later time frame. We trained the recommendation algorithms, using the rating data in the earlier time frame, and calculated the user-level predictive performance based on ratings provided later. Tables C1 and C2 in the Online Supplement present the results based on time-series analysis. The results are highly consistent with those reported earlier in Tables 3 and 4, suggesting that the findings are robust to the possible changes in users' ratings over time.

Second, we explored alternate accuracy metrics for user-level recommendation performance. Metrics such as MAE and RMSE are widely used in many real-world applications. Analyses that use these metrics allow us to test the generalizability of our findings. Tables C3 and C4 provide within-dataset and combined dataset regression results, respectively, using RMSE as the accuracy metric. Tables C5 and C6 provide corresponding results, using MAE as the accuracy metric. We find our results to be highly consistent across different accuracy metrics, meaning that the findings can be generalized to other accuracy measures.

7. Conclusion

7.1. Summary of Findings and Contributions

This study investigates the effects of users' rating characteristics on the user-level predictive accuracy of three recommendation algorithms, including item-based and user-based CF algorithms and the matrix factorization algorithm. We focused on three categories of rating characteristics: rating value (measured by rating mean and rating variance), rating structure (measured by density and the popularity mean of consumed items), and neighborhood network embeddedness (measured by eigenvector centrality and clustering coefficient).

Across five datasets from various application domains, our results from both within- and across-dataset analyses show that users' rating characteristics have consistent and significant effects on the user-level accuracy of recommendation algorithms. We find rating variance to be the most important factor in determining the predictive accuracy of recommendation algorithms. With other variables being equal, recommendation algorithms make more accurate predictions for users with smaller rating variance. The impact of rating variance is nonlinear and is more prominent when rating variance value is small, and then decreases as rating variance becomes larger.

Our results also show a positive effect of rating density, indicating that recommendation algorithms perform better for users with a higher number of ratings. In addition, the average popularity of consumed items has a positive effect on prediction accuracy, meaning that users who consume more popular items tend to have better prediction accuracy. Further, for neighborhood network embeddedness, we found that both eigenvector centrality and clustering coefficient have significant positive effects on predictive accuracy. That means users who are more embedded in the network, i.e., have more connections to others or share higher similarity in rating patterns with others, tend to receive more accurate predictions from recommender systems. Overall, the effects of the six rating characteristics on recommender systems' accuracy are highly consistent across different datasets and recommendation algorithms, providing evidence that our findings can be generalized to a variety of application settings.

Our research provides several contributions to the recommender systems literature. To the best of our knowledge, this study is the first to provide a structured investigation of the impacts of users' rating characteristics on recommendation algorithms' performance at the individual-user level. The findings of this study help to enrich our understanding of the relationships between rating data characteristics and the predictive accuracy of recommendation algorithms.

Another important contribution of this work is to introduce and verify the feasibility of implicit networks in the recommender systems context. We present an approach to construct an implicit network based on users' co-rating patterns, and explore the value of using an implicit network to predict accuracy performance. When there is no explicit social relation established or observable in the system, we show that implicit networks, such as the neighborhood network, can be constructed based on users' co-rating similarities. Our analysis also demonstrates the value of using an implicit network to predict accuracy performance. The results suggest that network embeddedness is positively related to predictive performance for recommendation algorithms. Therefore, recommender system designers could potentially apply network analysis techniques on implicit networks to improve system performance.

Further, our findings provide important practical implications for the design of recommender systems as well as a useful regression model that designers can apply to estimate recommendation algorithms' performance for individual users, using only six simple rating characteristics variables. This would allow designers to diagnose potential issues for users who receive low-quality predictions proactively and then customize the systems design for these users accordingly (e.g., by using an alternative algorithm, by inducing more favorable rating distributions). Our results indicate that some variables are more important than are other variables, allowing system designers to prioritize item recommendations shown to users to maximize system performance. We further discuss practical implications in the next section.

7.2. Implications of Findings

We provide several example approaches of how our proposed model can be utilized to improve the design of recommender systems. First, using the proposed model, system designers can predict recommendation algorithms' performance and decide which one to use for every individual user. In our experiment, we investigated how well our model can identify the best performing algorithm for each individual user correctly. Specifically, we used the regression models to predict the best algorithm for each user and compared the prediction with the actual best performing algorithm for the same user. Table 5 provides a summary of the precision, recall, and F1 scores of our model in correctly predicting the best performing algorithms. Because the classification has three labels (i.e., algorithms), we calculated the precision, recall, and F1 scores for each label separately. Overall, the average precision value across five datasets is 0.772, indicating that, when our model identifies an algorithm as the best performing one for a user, 77.2% of the time, this algorithm is truly the best one. In the meantime, the average recall value across five datasets is 0.883, indicating that, when an algorithm outperforms others, our model can correctly identify 88.3% of these cases. The average F1 score across five datasets is 0.823. Such results suggest that our model is capable of choosing the best performing algorithm for most users.

Table 5 Accuracy Performance in Identifying the Best Performing Algorithm for Each Individual User.

Dataset	Item-based CF			User-based CF			FunkSVD		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
MovieLens 100K	0.8680	0.9767	0.9192	0.7742	0.9025	0.8334	0.8198	0.9301	0.8715
MovieLens 1M	0.8773	0.9951	0.9325	0.7644	0.8710	0.8143	0.7336	0.8586	0.7912
Netflix	0.8813	0.9964	0.9353	0.7646	0.8506	0.8053	0.7350	0.8321	0.7806
Yahoo! Music	0.8255	0.9800	0.8961	0.7561	0.9006	0.8220	0.5878	0.6274	0.6069
Yelp	0.6616	0.7435	0.7002	0.6692	0.7868	0.7233	0.8572	0.9905	0.9191

Second, the system designers can employ our model to identify users who are difficult to predict and set them aside to use alternative approaches to generate recommendations for these users. For example, the system can suggest most-liked items (i.e., items with highest average ratings) to these users and predict these users' preferences on most-liked items, using heuristics approaches, such as item average and user average ratings (Koren 2010). Our experiments show that simple item average-based predictions can lead to higher accuracy for these most-liked items than can popular personalized algorithms. In other words, we show that our model can be used for a "preprocessing step" to identify difficult-to-predict users and can handle these users separately with alternative recommendation approaches.

Specifically, in our experiment, we first identified 20% of the users who are most difficult to predict by ranking the predicted wRMSE based on our model. Then we recommended the top 50 items with highest average rating values to each user. We calculated predictions on these items based on item-average algorithm. We then compared the accuracy performance (as calculated as wRMSE) of the item-average algorithm with that of the well-known personalized algorithms, including item-based and user-based CF and FunkSVD approaches. Table 6 presents our results for predictive accuracy. It is evident that, for those difficult-to-predict users, recommending popular items provided a significant smaller wRMSE than did recommendations based on the three personalized algorithms. To ensure the robustness of this implication, we also applied our approach to identify 10% and 30% of users who are difficult to predict and found our results to be highly consistent.

By comparing rating characteristics of these difficult-to-predict users with the other users (see Table B6 in the Online Supplement), we find that the rating variance of these users is significantly larger, while the rating density and Eigenvector centrality are significantly smaller than for the other users. Our results show that, for users who have rated few items, have high variations in ratings, and are not well-connected to the other users, personalized algorithms cannot provide accurate predictions and thus non-personalized recommendation should be implemented.

Table 6 Comparison of wRMSE for Personalized vs. Item-average Algorithms on Highest-liked Items for Difficult-to-Predict Users.

Dataset	ItemCF vs. HighlyRated			UserCF vs. HighlyRated			FunkSVD vs. HighRated		
	ItemCF	ItemAvg	Δ	UserCF	ItemAvg	Δ	SVD	ItemAvg	Δ
MovieLens 100K	1.1979	1.0345	0.1635***	1.2170	1.0346	0.1825***	1.2057	1.0308	0.1749***
MovieLens 1M	1.1085	0.9891	0.1195***	1.1442	0.9928	0.1514***	1.1236	0.9909	0.1327***
Netflix	1.1513	0.8519	0.2994***	1.1853	0.8462	0.3391***	1.1823	0.8419	0.8419***
Yahoo! Music	1.5845	0.7336	0.7336***	1.6194	1.6194	0.8836***	1.6949	0.7319	0.9629***
Yelp	1.2737	0.3274	0.9463***	1.2461	0.3262	0.9199***	1.2335	0.3272	0.9062***

Note: $^{***}p \leq 0.001$

Third, the system designers can use our model as a tool for an intelligent sampling of a rating dataset. For example, the model can be used to filter the bottom difficult-to-predict users and to retain the remaining users (i.e., a subsample of the original user population) in the dataset. In our experiment, we empirically show that the intelligent sampling step could improve the overall recommendation performance for these remaining users; albeit, only a subsample of the data was used. Intuitively, sampling leads inherently to the loss of information, but we show that our model can help to filter the noise in the original data and, therefore, help to achieve better recommendation performance on the remaining dataset. Specifically, in our experiment, we first identified 20% of the users who are predicted to have the largest wRMSE values based on our model. We then removed these users and ran the three algorithms (i.e., item-based CF, user-based CF, and FunkSVD) on the remaining 80% of the users 30 times. We compare the wRMSE performance on the subsample with the original wRMSE (Table 7). Our results show that the subsample yields significantly higher accuracy performance than does the original dataset.

Table 7 Comparison of wRMSE for Original Datasets and Extracted Samples with 80% Users.

Dataset	Item-based CF			User-based CF			FunkSVD		
	Original	Sample	Δ	Original	Sample	Δ	Original	Sample	Δ
MovieLens 100K	0.9081	0.8415	0.0666***	0.9169	0.8461	0.0708***	0.9108	0.8379	0.0728***
MovieLens 1M	0.8525	0.7927	0.0598***	0.8673	0.8028	0.0645***	0.8708	0.8060	0.0648***
Netflix	0.8699	0.7976	0.0723***	0.8841	0.8080	0.0761***	0.8887	0.8123	0.0764***
Yahoo! Music	1.1122	1.0689	0.0866***	1.1641	1.0743	0.0897***	1.2019	1.1023	0.0996***
Yelp	0.9567	0.8683	0.0884***	0.9485	0.8653	0.0832***	0.9254	0.8448	0.0806***

Note: $^{***}p \leq 0.001$.

7.3. Future Work

This paper presents the initial step for studying the impacts of individual-users' rating characteristics on the user-level accuracy performance of recommendation algorithms. There are a few limitations of this research, and additional work is needed to provide a more comprehensive understanding of the issue. First, this research focuses on explicit rating data. Users' ratings do not fully capture all the items consumed by users (e.g., a user might consume some items but never rate them). One direction in which to extend this work is to explore other features that are not based on explicit numeric ratings. Examples include the study of users' demographic information, consumption behaviors, and implicit feedback observed by the system (e.g., users' movie streaming duration and browsing behaviors). Such non-rating features could provide complementary information about users. Future research should explore how the non-rating feedback influences the user-level recommendation performance of popular recommendation algorithms.

Second, this study uses the aggregated prediction errors (e.g., weighted RMSE, RMSE, MAE) for measuring recommendation algorithms' performance, and the estimation of regression model focuses on minimizing these aggregated errors. Such estimators, however, do not minimize the worst possible prediction performance for an individual user. Future research needs to explore other approaches such as the minimax estimator that minimizes the upper bound of prediction errors for each user. In addition, future research needs to explore other aspects of recommendation performance, such as diversity, novelty, and coverage. It has been recognized, in both research and practice, that good recommendations should be not only accurate but also diverse, novel, and serendipitous. Future work can explore how users' rating characteristics affect these important performance metrics of recommendation algorithms.

Third, our use of observational rating data collected in real-world systems constitutes another limitation of this work. When users provide ratings, their ratings may be influenced by many factors other than their true preferences. Examples include users' current mood, time of the day, input interfaces (e.g., mobile, computer), and satisfaction with the platform (e.g., delivery of the item). Rating data also can be skewed, with more positive ratings than negative ratings. This is because users tend to consume (and later rate) the items that are expected to fit their preferences. The skewed distribution of users' feedback data (both implicit and explicit) is a common issue in the empirical research of recommender systems. To address this issue, a field experiment would be useful to control for the items displayed to and consumed by users and to collect user feedback on items. This constitutes another direction for future research.

References

Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE TKDE* 17(6):734–749.

Adomavicius G, Zhang J (2012) Impact of data characteristics on recommender systems performance. *ACM Transactions on Management Information Systems* 3(1):3.

Allen B (2017) The secrets of netflixs recommendation system and why it may not work for you. `http://www.radiotimes.com/news/2017-09-08/netflix-reccomendation-how-does-it-work/`.

Bell RM, Koren Y (2007) Improved neighborhood-based collaborative filtering. *KDD Cup*, 7–14.

Bennett J, Lanning S, Netflix N (2007) The netflix prize. *KDD Cup*.

Berkovsky S, Freyne J (2015) Web personalization and recommender systems. *Conference on Knowledge Discovery and Data Mining*, 2307–2308 (ACM).

Chee SHS, Han J, Wang K (2001) Rectree: An efficient collaborative filtering method. *International Conference on Data Warehousing and Knowledge Discovery*, 141–151.

Cosley D, Lam SK, Albert I, Konstan JA, Riedl J (2003) Is seeing believing? how recommender system interfaces affect users' opinions. *SIGCHI Conference on Human Factors in Computing Systems*, 585–592, CHI '03 (New York, NY, USA: ACM).

Deshpande M, Karypis G (2004) Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)* 22(1):143–177.

Ekstrand MD, Ludwig M, Konstan JA, Riedl JT (2011) Rethinking the recommender research ecosystem: Reproducibility, openness, and lenskit. *ACM Conference on Recommender Systems*, 133–140.

Funk S (2006) Netflix update: Try this at home. URL `http://sifter.org/~simon/journal/20061211.html`, accessed: 08/10/2016.

Grewal R, Lilien GL, Mallapragada G (2006) Location, location, location: How network embeddedness affects project success in open source systems. *Management Science* 52(7):1043–1056.

Harper FM, Konstan JA (2016) The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 5(4):19.

Herlocker JL, Konstan JA, Riedl J (2000) Explaining collaborative filtering recommendations. *ACM Conference on Computer Supported Cooperative Work*, 241–250.

Herlocker JL, Konstan JA, Terveen LG, Riedl JT (2004) Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)* 22(1):5–53.

Hsu CN, Chung HH, Huang HS (2004) Mining skewed and sparse transaction data for personalized shopping recommendation. *Machine Learning* 57(1-2):35–59.

Hu Y, Koren Y, Volinsky C (2008) Collaborative filtering for implicit feedback datasets. *IEEE International Conference on Data Mining*, 263–272.

Huang Z, Zeng DD (2011) Why does collaborative filtering work? transaction-based recommendation model validation and selection by analyzing bipartite random graphs. *INFORMS Journal on Computing* 23(1):138–152.

Jawaheer G, Szomszor M, Kostkova P (2010) Comparison of implicit and explicit feedback from an online music recommendation service. *HetRec Workshop*, 47–51.

Jonnalagedda N, Gauch S (2013) Personalized news recommendation using twitter. *Web Intelligence (WI) and Intelligent Agent Technologies*, 21–25.

Kadushin C (2012) *Understanding Social Networks: Theories, Concepts, and Findings* (OUP USA).

Kluver D, Konstan JA (2014) Evaluating recommender behavior for new users. *ACM Conference on Recommender Systems*, 121–128.

Konstan JA, Miller BN, Maltz D, Herlocker JL, Gordon LR, Riedl J (1997) Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM* 40(3):77–87.

Koren Y (2010) Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 4(1):1.

Koren Y, Bell R, Volinsky C, et al. (2009) Matrix factorization techniques for recommender systems. *Computer* 42(8):30–37.

Lee C, Pham M, Jeong MK, Kim D, Lin DKJ, Chavalitwongse WA (2015) A network structural approach to the link prediction problem. *INFORMS Journal on Computing* 27(2):249–267.

Li G, Chen Q (2016) Exploiting explicit and implicit feedback for personalized ranking. *Mathematical Problems in Engineering* 2016.

Liu NN, Meng X, Liu C, Yang Q (2011) Wisdom of the better few: Cold start recommendation via representative based rating elicitation. *ACM Conference on Recommender Systems*, 37–44.

Mason WA, Conrey FR, Smith ER (2007) Situating social influence processes: Dynamic, multidirectional flows of influence within social networks. *Personality and Social Psychology Review* 11(3):279–300.

Miller BN, Konstan JA, Riedl J (2004) Pocketlens: Toward a personal recommender system. *ACM Transactions on Information Systems (TOIS)* 22(3):437–476.

Pan R, Zhou Y, Cao B, Liu NN, Lukose R, Scholz M, Yang Q (2008) One-class collaborative filtering. *ICDM*, 502–511 (IEEE).

Pu P, Chen L, Hu R (2012) Evaluating recommender systems from the users perspective: survey of the state of the art. *User Modeling and User-Adapted Interaction* 22(4):317–355.

Ransbotham S, Kane GC, Lurie NH (2012) Network characteristics and the value of collaborative user-generated content. *Marketing Science* 31(3):387–405.

Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J (1994) Grouplens: An open architecture for collaborative filtering of netnews. *ACM Conference on Computer Supported Cooperative Work*, 175–186.

Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. *Conference on World Wide Web*, 285–295.

Sarwar BM, Konstan JA, Borchers A, Herlocker J, Miller B, Riedl J (1998) Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. *Conference on Computer Supported Cooperative Work*, 345–354.

Shani G, Gunawardana A (2011) Evaluating recommendation systems. *Recommender Systems Handbook*, 257–297 (Springer).

Ting I, Yu PL, et al. (2016) Measuring the effect of social network data on music recommendation. *International Knowledge Management in Organizations Conference*, 24.

Wasserman S, Faust K (1994) *Social Network Analysis: Methods and Applications*, volume 8 (Cambridge University Press).

Yan L, Peng J, Tan Y (2015) Network dynamics: How can we find patients like us? *Information Systems Research* 26(3):496–512.

Yang X, Steck H, Guo Y, Liu Y (2012) On top-k recommendation using social networks. *ACM Conference on Recommender Systems*, 67–74.

Zhou X, Xu Y, Li Y, Josang A, Cox C (2012) The state-of-the-art in personalized recommender systems for social networking. *Artificial Intelligence Review* 37(2):119–132.