Improving the Quality of Predictions using Textual Information in Online User Reviews

Gayatree Ganu*, Yogesh Kakodkar, Amélie Marian

Department of Computer Science, Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ - 08854, USA

Abstract

Online reviews are often accessed by users deciding to buy a product, see a movie, or go to a restaurant. However, most reviews are written in a free-text format, usually with very scant structured metadata information and are therefore difficult for computers to understand, analyze, and aggregate. Users then face the daunting task of accessing and reading a large quantity of reviews to discover potentially useful information. We identified topical and sentiment information from free-form text reviews, and use this knowledge to improve user experience in accessing reviews. Specifically, we focus on improving recommendation accuracy in a restaurant review scenario. We propose methods to derive a text-based rating from the body of the reviews. We then group similar users together using soft clustering techniques based on the topics and sentiments that appear in the reviews. Our results show that using textual information results in better review score predictions than those derived from the coarse numerical star ratings given by the users. In addition, we use our techniques to make fine-grained predictions of user sentiments towards the individual topics covered in reviews with good accuracy.

Keywords: Text classification, user reviews, social information filtering, personalized recommendations, probabilistic clustering

1. Introduction

In 2006, Time Magazine chose as its *Person of the Year* the millions of anonymous contributors of user-generated content. Today, Web users have whole-heartedly incorporated peer-authored posts into their daily decisions. Despite the growing popularity, there has been little research on the quality of the content. In addition, web sites providing user reviews are surprisingly technolog-

*Corresponding author

Email addresses: gganu@cs.rutgers.edu (Gayatree Ganu), yogeshk@cs.rutgers.edu (Yogesh Kakodkar), amelie@cs.rutgers.edu (Amélie Marian) ically poor: users often have no choice but to browse through massive amounts of text to find a particular piece of relevant information.

Accessing and searching text reviews is particularly frustrating when users only have a vague idea of the product or its features and they need a recommendation or closest match. Keyword searches typically do not provide good results, as the same keywords routinely appear in good and in bad reviews [1]. Another challenge in understanding reviews is that a reviewer's overall rating might be largely reflective of product features in which the search user is not interested. Consider the following example:

EXAMPLE 1: *The New York City restaurant Lucky Cheng's in* Citysearch *(http://newyork.citysearch.com) has 65 user reviews of which 40 reviews have a 4 or 5 star rating (out of 5 possible stars). Most positive reviews, however, praise the ambience of the restaurant, as shown in the following sentences:*

- "obviously it's not the food or drinks that is the attraction, but the burlesque show"
- "The food was okay, not great, not bad.[...]Our favorite part, though, was the show!"

The negative reviews complain at length about the price and service. Users not interested in ambience would probably not want to dine at this restaurant. However, a recommendation using star ratings alone would not be able to harness such user-specific preferences towards restaurant features.

Ideally, users should not have to read through several reviews, but should be presented with items that they would find interesting or useful based on some notion of preference through similarity with other users or items. This task of preference matching is carried out by recommendation systems [5]. Current recommendation systems such as the ones used by Netflix or Amazon [24] rely predominantly on structured metadata information to make recommendations, often using only the star ratings, and ignore a very important information source available in reviews: the textual content.

We propose techniques that harness the rich information present in the body of the reviews by identifying the review parts pertaining to different product features (e.g., food, ambience, price, service for a restaurant), as well as the sentiment of the reviewer towards each feature (e.g., positive, negative or neutral) and leverage this information to improve user experience. Identifying such structured information from free-form text is a challenging task as users routinely enter informal text with poor spelling and grammar. We performed an in-depth classification of a real-world restaurant review data set and report on our techniques and findings. Our work addresses categorization and sentiment analysis at the sentence level as web reviews are short and designed to convey detailed information in a few sentences. We apply our text analysis to a recommendation scenario and show that the rich textual information can improve rating prediction quality. In addition, we propose methods to predict the sentiments of users towards individual restaurant features and enhance user experience by presenting the review parts pertaining to these features.

Our work, performed as part of the **URSA** (User Review Structure Analysis) project, takes the novel approach of combining natural language processing, machine learning and collaborative filtering to harness the wealth of detailed information available in web reviews. Our techniques utilize the free form textual data from user reviews for collaborative filtering, a domain where most studies have focused on using ratings and other structured metadata. In [12], our preliminary results indicated that using text for rating prediction was a promising direction. In the current paper, we present personalized recommendation techniques that use the full text of a review to make ratings predictions as well as predictions on user sentiment towards restaurant features. In particular we make the following novel contributions:

- We implement a new quadratic regression model using all of the detailed textual information obtained from the text classification, to derive text-based ratings (Section 3.2). In comparison with the simple ad-hoc and linear regression presented in [12], the quadratic model is a better fit for our data (estimated by the lowered error in regression), and yields more accurate rating predictions.
- We compare the predictive power of star and textual ratings using average-based strategies that incorporate the rating behavior of the user, the average quality of the restaurant, and a combination of both (Section 3.3).
- We reviewed state of the art recommendation techniques and evaluated their performance on our restaurant reviews corpus. As described in Section 4.1, rating-based methods using latent factorization and neighborhood models do not yield significant improvements over average-based baseline predictions for our sparse dataset.
- We utilize the rich textual information present in the reviews to better group similar users for making recommendations. Users who have reviewed common restaurants are clustered together if they have liked or disliked the same aspects of the restaurant in the text of their reviews, thus providing an approach to address the problem outlined in Example 1. We implement a text-based soft clustering of users and design a novel prediction approach for making personalized predictions in Section 4.2.
- We present an approach to predicting not just a numeric rating, but the sentiments of users towards individual restaurant features (Section 5).

This paper is structured as follows. We describe our restaurant reviews data set in Section 2, discuss our text classification approach and describe the evaluation settings for our prediction experiments. In Section 3, we propose new regression-based measures that take into account the textual component of reviews for deriving alternate text-based ratings for user reviews. We then turn our focus to accurately predicting ratings for making useful recommendations, using average-based recommendation strategies. In Section 4, we evaluate popular rating-based methods like matrix factorization and KNN and evaluate the use of the textual information for clustering like-minded users in personalized prediction settings. We show that relying on user reviewing behaviors, as determined by the type of sentences covered in the reviews, results in an improvement in predictions over techniques that only consider ratings. We then use the textual information to predict user sentiments towards individual restaurant features in Section 5. We report on related work in Section 6 and conclude in Section 7.

2. Data Description

We first describe the restaurant reviews data set (Section 2.1) and our methods for harnessing the rich textual information via classification of review sentences with topical and sentiment information (Section 2.2). We then outline our ratings prediction evaluation settings in Section 2.3.

2.1. ***Restaurant Review Data Set***

We focused our classification effort on a restaurant review data set, extracted from the NY Citysearch web site. The corpus contains 5531 restaurants and 51162 reviews; reviews contain structured metadata (star rating, date) along with the text. Typically reviews are small; the average user review has 5.28 sentences. The reviews are written by 31814 distinct users, for whom we only have unique username information.

The data set is sparse: users typically review only a few restaurants. Over 25,000 users have written only one review. This sparsity is a typical problem in collaborative filtering systems and is detrimental to the success of several recommendation algorithms.

2.2. ***Classification of Reviews***

Web reviews have a combination of linguistic characteristics that depart from the genres traditionally considered in the field of information processing: the language is often quite specific to a particular domain (reviewers of electronic goods, for instance, use many technical terms to describe product features like resolution, battery life, zoom); at the same time reviews are unedited and often contain informal and ungrammatical language. Reviews also contain anecdotal information, which does not provide useful, or usable, information for the sake of automatic quality assessment.

Our approach to addressing these challenges is to consider a review not as a unit of text, but as a set of sentences, each with their own topics and sentiments. This added structural information provides valuable information on the textual content at a fine-grain level. We model our approach as a multi-label text classification task for each sentence where labels are both about topics and sentiments. We analyzed the data to identify the following six topics specific to the restaurant reviews domain: Food, Service, Price, Ambience, Anecdotes, and

Sentence Category	**Accuracy**	**Precision**	**Recall**
FOOD	84.32	81.43	76.72
SERVICE	91.92	81.00	72.94
PRICE	95.52	79.11	73.55
AMBIENCE	90.99	70.10	54.64
ANECDOTES	87.20	49.15	44.26
MISCELLANEOUS	79.40	61.28	64.20
Sentiment	**Accuracy**	**Precision**	**Recall**
POSITIVE	73.32	74.94	76.60
NEGATIVE	79.42	53.23	45.68
NEUTRAL	80.86	32.34	23.54
CONFLICT	92.06	43.96	35.68

Table 1: 7-Fold cross validation of classifier results.

Miscellaneous. In addition to the topics, sentences have an associated sentiment: Positive, Negative, Neutral, or Conflict.

The details of our classification efforts are described in [13]. The classification involved manually annotating a small classifier training set of review sentences, and then training SVM classifiers for the automatic sentence annotation of the entire corpus.

2.2.1. Manual Sentence Annotation

Classifying text along different topics and sentiments is a notably challenging task [13]. To classify sentences into the above mentioned categories and sentiment classes, we manually annotated a training set of approximately 3400 sentences with both category and sentiment information. To check for agreement, 450 of these sentences were annotated by three different annotators. The kappa coefficient (K) measures pairwise agreement among a set of annotators making category judgments, correcting for expected chance agreement [29]. A Kappa value of 1 implies perfect agreement, the lower the value, the lower the agreement. The inter-annotator agreements for our annotations were very good (Kappa above 0.8) for the Food, Price, and Service categories and Positive sentiment. The Negative sentiment (0.78), Neutral and Conflict sentiments, Miscellaneous and Ambience categories all had good agreements (above 0.6). The ambiguous Anecdotes category is the only one for which the Kappa value was moderate (0.51).

2.2.2. Automatic Sentence Classification

We trained and tested Support Vector Machine classifiers [18] on our manually annotated data (one classifier for each topic and one for each sentiment type). Features for all classifiers were stemmed words (experiments did not suggest significant improvements in accuracy when other features like n-grams and semantic features were used for classification). We used svm light[1] with default

[1]`http://svmlight.joachims.org`

parameters, although experiments with SVM-HMM, Naive Bayes, Bayes Net and Decision Tree classifiers were also conducted.

We performed 7-fold cross validation [20] and used accuracy, precision and recall to evaluate the quality of our classification (see Table 1). Precision and recall for the main categories of Food, Service and Price and the Positive sentiment were high (70%), while they were lower for the Anecdotes, Miscellaneous, Neutral and Conflict categories. These low results could be due to the ambiguous nature of these categories but also due to the small amount of training instances in our corpus for these categories in particular.

While the specific categories we identified are tailored for a restaurant scenario, our classification approach could easily be translated to other types of data sets after a topical analysis to identify product-specific sentence categories.

2.3. ***Evaluation Setting***

To evaluate the predictive value of our recommendation methods, we randomly extracted three test sets of around 260 reviews each from the restaurant data set; the remaining reviews comprised the corresponding training sets. A review set aside in the test set is not used in making predictions, but is only used in evaluating the accuracy of the predictions. For personalized recommendations, we are interested in using user-specific information for clustering users, and we need at least one review written by the users in the test set to derive user-specific information. Therefore, two of our test sets – A and B, are randomly chosen such that each test user has at least one review in the training set in addition to the review set aside for the test set.

Test set C contains one review each from users who have rated at least 5 restaurants. Therefore, Test set C contains more usable user-specific information than the randomly chosen Test sets A and B.

Our data contains only the review date information and no time stamp. A majority (86%) of users have written all their reviews on the same day. Hence, we are unable to create test sets containing the last review written by the user, as is often done, e.g. the Netflix Challenge test set [4].

We now focus on predicting ratings for the test set reviews using baseline average-based strategies.

3. Predicting Restaurant Ratings

In this section, we first describe our methodology (Section 3.1) for predicting the overall ratings for the reviews in the test sets. To compare the use of star ratings with the textual data in a recommendation scenario, we propose a novel method for deriving textual ratings using our sentence classification in Section 3.2. Textually derived ratings serve as an alternate assessment in the review based on the user sentiment towards different product aspects. We then evaluate the predictive utility of the star ratings and the text-based ratings using average-based prediction strategies in Section 3.3. Note that in Section 5, we go beyond the goal of predicting the overall review rating and focus on making fine-grained predictions on user sentiment towards individual restaurant aspects.

3.1. ***Methodology***

Our goal is to use the information present in the training data to accurately predict the ratings in the test set. To explore whether the text in reviews is a better predictor of user assessment of a restaurant than the star ratings, we derive an alternate textual rating from the body of the reviews as described in Section 3.2. Using this analysis, we have two alternate methods to manipulate the information present in the training data: the star ratings in the reviews, and the textual ratings derived from the body of the reviews.

In addition, the reviews in the test set also contain both star ratings and textual ratings. Therefore, we have two prediction goals: accurately predicting the star ratings of the test set reviews and accurately predicting their text ratings.

We use the popular root mean square error (RMSE) accuracy metric to evaluate our prediction techniques [15].

3.2. ***Textually Derived Ratings***

The text of a review (as approximated by its associated topics and sentiments) can enable us to capture the detailed assessment by a user of the restaurant. We use a regression-based method for deriving textual ratings from the review text as described in the following section.

3.2.1. Regression-based Method

Typically, users assign different degrees of importance to the topics of their reviews. For each review in the corpus, we propose a textual rating which incorporates topics and sentiments with varying levels of importance into a regression-based rating. Regression allows us to learn weights to be associated with each sentence type. These weights are learned from the data set itself, and therefore closely represent how people write reviews in a domain. Our regression models the user-provided star ratings as the dependent variable; the sentence types represented as *(topic,sentiment)* pairs are the independent variables, i.e., we performed a multivariate regression which learns weights or importance to be associated with the different textual information (represented by the several sentence type variables).

We computed the multivariate regression using the least squares estimates method. We performed a qualitative comparison between different sentence types settings and varying regression models, as described in the following section.

3.2.2. Four-Sentiment Second-Order Regression

A important step when fitting data is to find a good regression model. We experimented with linear multivariate models, as well as second order and third order models. The goodness of fit for these models is estimated using the root mean squared error for the regression [26].

We observed that the quadratic regression model, incorporating all the textual features (six topics and four sentiments), is a better fit for the restaurant

Constant	3.68			
1st **Order Variables**	Positive	Negative	Neutral	Conflict
Food	2.62	-2.65	-0.078	-0.690
Price	0.395	-2.12	-1.27	0.929
Service	0.853	-4.25	-1.83	0.358
Ambience	0.747	-0.269	0.162	0.215
Anecdotes	0.957	-1.75	0.061	-0.186
Miscellaneous	1.30	-2.62	-0.303	0.358
2nd **Order Variables**	Positive	Negative	Neutral	Conflict
Food	-2.00	2.04	-0.134	0.664
Price	-0.265	2.03	2.26	-1.01
Service	-0.526	3.15	1.79	0.354
Ambience	-0.438	0.801	-0.263	-0.595
Anecdotes	-0.401	1.97	-0.081	-0.262
Miscellaneous	-0.651	2.38	0.492	-0.089

Table 2: Four-sentiment regression weights.

reviews data set than the earlier proposed linear model in [12] (as estimated by the lowered error in regression). Our quadratic regression model for deriving textual ratings has the general form for three independent variables shown in Equation 1.

$$y \pm \phi = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_1^2 + \beta_5 x_2^2 + \beta_6 x_3^2 \qquad (1)$$

In the above, β_0, $\beta_1, \ldots, \beta_6$ are the unknown weights that we wish to determine. x_1, x_2, x_3 are the sentence types frequencies. The dependent variable y is the star rating. ϕ is the error in regression, a good model will have a low error.

This model uses all information derived from the text classification which is beneficial for building a robust system. We build our model on the 50K examples in the training set as described in Section 2.3. Note that our model provides a regression constant which serves as a default rating when no textual information is available. The constant of 3.68 is slightly skewed towards a good review (star rating 4 or 5); this is consistent with the distribution of star ratings in the restaurant review corpus as discussed in [13]. Finally, the second-order weights (shown in Table 2) have the reverse polarity as the corresponding first-order weights: the second order variables tend to dampen the effects of the first-order variables if many sentences of a type are present in a review.

The weights for our quadratic regression model are shown in Table 2. The proportion of Positive and Negative sentiment sentences have a clear effect on the rating in a review, as shown by the highly polar regression weights for these sentiments. As expected, the weights confirm that the Food category has the highest impact on the perception of a restaurant. The weights of the negative Price and Service related sentences are quite significant, indicating that unacceptable prices or poor service in a restaurant has a very adverse impact on the dining experience of users.

3.3. ***Average-based Predictions***

We now focus on making predictions for the reviews in our test sets, using three average-based techniques. Our methods use the average assessment of

		Restaurant Average			User Average			**Combined**		
		TestA	**TestB**	**TestC**	TestA	TestB	TestC	TestA	TestB	TestC
Predicting Star rating	Using Star rating	**1.127**	**1.267**	**1.126**	1.313	1.349	1.061	1.283	1.363	1.095
	Using Textual rating	**1.126**	**1.224**	**1.046**	1.149	1.231	1.035	1.143	1.236	1.029
Predicting Textual rating	Using Star rating	**0.703**	**0.718**	**0.758**	0.971	0.969	0.649	0.990	1.031	0.812
	Using Textual rating	**0.545**	**0.557**	**0.514**	0.603	0.631	0.491	0.609	0.637	0.523

Table 3: Prediction RMSE using average-based methods.

the restaurant, the average rating behavior of the user, and a combination of both. For each strategy, predictions using text ratings provide better predicting accuracy (lower RMSE values) as compared to the predictions using the star ratings as shown in Table 3.

In the restaurant average-based prediction technique the rating of a test review is predicted as the average rating of all the other reviews for the test restaurant. The resulting RMSE values are shown in the leftmost columns of Table 3. For the task of predicting star ratings, there is a significant improvement in prediction accuracy (7.1% and 3.4%) achieved for Test sets B and C, when textual ratings are used for making predictions.

For predicting textual ratings, the text again always outdoes the star ratings in making accurate predictions. Textual ratings indicate the general preference of a user towards the restaurant. However, information in the review about the sentence topics and sentiments are combined in a single rating. Predicting text ratings coarsely predicts the textual component of a review but does not predict individual topics and sentiments that are likely to be in the review. We will focus on such detailed qualitative predictions in Section 5. Note that the textual ratings have a lower standard deviation and therefore average-based strategies for predicting text ratings are expected to have lower errors.

We next examine the user average-based prediction strategy where the predicted value is the average rating of all the other reviews written by the test user (second column of Table 3). Lastly, we use a combination method where the predicted rating uses the deviation of the user average and the restaurant average from the data set average rating, as suggested in [21]. The results for this combined average-based method are included in the rightmost columns of Table 3. For Test set C, where users have reviewed many restaurants, user average or combined average prediction strategies prove to be less erroneous than the aforementioned restaurant average strategy. However, a large majority of users do not write many reviews (78% users have written only one review). The restaurant average predictions performs better in the generalized setting. Thus, we use the restaurant average approach as our baseline.

The results in Table 3 show that for each of the three average-based prediction strategies, using our textual ratings has a considerable advantage for making accurate predictions over the star ratings. We now focus on making better predictions using personalized recommendation strategies by finding like-minded users.

4. Personalized Rating Prediction

A limitation of the prediction metrics presented in the previous section is that they do not take advantage of all the usable information: the restaurant average prediction strategy results in all users receiving the same prediction for a restaurant regardless of individual preferences. In order to make better and personalized predictions there is a need to leverage information beyond the restaurant average by taking into account the similarities between users.

In this section, we investigate personalized recommendation techniques. In Section 4.1, we implement two popular state of the art collaborative filtering methods that rely on ratings (either star ratings or textually derived scores) for making predictions. In Section 4.2 we demonstrate the utility of our sentence classification for making accurate recommendations. We not only use textually derived ratings, but also utilize the textual patterns in user reviews for a grouping or clustering of similar users using a text-based soft clustering of users.

4.1. ***Rating-based Personalized Prediction***

In recent years, there have been several studies on collaborative filtering models that rely predominantly on the ratings given by the users to the different items to make predictions. Such models saw a surge in popularity during the Netflix challenge [4]. In this section, we implement two ratings-based methods for making personalized predictions. In Section 4.1.1, we first implement a factorization method on the matrix of ratings in the training set to uncover latent features for predicting the ratings in the test sets. Next, in Section 4.1.2 we implement a neighborhood-based model for grouping or clustering of similar users.

4.1.1. Latent Factor Model

Matrix factorization (MF) has been useful for collaborative filtering in several previous studies [3, 2, 32, 35, 22] due to its ability to discover latent factors underlying the ratings given by the users to the items. These latent factors can then be used to predict unknown ratings. For a $m \times n$ matrix R comprising ratings given by m users to n restaurants, MF approximates R with the best rank-k approximation $\hat{R_k}$. $\hat{R_k}$ is computed as the product of two matrices $P_{m\times k}$ and $Q_{n\times k}$. In other words, to approximate R we factorize it into two low dimensional matrices P and Q (typically $k << min(m,n)$) such that $\hat{R_k} = PQ^T$ or $R \approx PQ^T$.

MF associates each user i with a user-factors vector P_i of size k representing the underlying latent factors explaining user ratings, similarly each restaurant j with a vector Q_j. To find the suitable factors P and Q we apply a gradient descent method [32, 22]. We start with randomly initializing $P_{m\times k}$ and $Q_{n\times k}$, and calculate how different their product $\hat{R_k}$ is from R for the known ratings. Note that R is a very sparse matrix, with zeros representing missing or unknown ratings. Let (i,j) represent the ℓ known ratings in the dataset given by users i to restaurants j. The basic form of the squared approximation error is computed as follows:

		TestA	TestB	TestC
Predicting Star Rating	Using Star rating	1.187	1.270	1.146
	Using Textual rating	1.148	1.215	1.083
Predicting Textual rating	Using Star rating	0.856	0.913	0.838
	Using Textual rating	0.630	0.640	0.599

Table 4: Prediction RMSE using matrix factorization for personalized predictions based on ratings.

$$\begin{aligned} {e_{ij}}^2 &= (R_{ij} - P_i {Q_j}^T)^2 \qquad for(i,j) \in \ell \\ &= \sum_{(i,j)\in\ell} \left(r_{ij} - \sum_k p_{ik} q_{kj} \right)^2 \end{aligned} \tag{2}$$

To avoid over fitting, we apply regularization to the basic form [3, 32] by penalizing with the magnitude of the user vector P_i and restaurant vectors Q_j. We introduce the regularization parameter λ which controls the magnitude of the vectors P_i and Q_j such that they would be a good approximation of R without containing large numbers. Therefore, the error is computed as:

$$e'_{ij} = \frac{1}{2}\left({e_{ij}}^2 + \lambda(\|P_i\|^2 + \|Q_j\|^2) \right) \tag{3}$$

We iteratively reduce the error in Equation 3 by implementing a gradient descent method to find a local minimum on the error. We compute the gradient of e'_{ij} for each k as follows:

$$\frac{\partial}{\partial p_{ik}} e'_{ij} = -e_{ij}.q_{kj} + \lambda.p_{ik}, \frac{\partial}{\partial q_{kj}} e'_{ij} = -e_{ij}.p_{ik} + \lambda.q_{kj} \tag{4}$$

Therefore, in each iteration we change the values in P and Q to decrease the approximation error. The change in the values is in small steps controlled by α, as follows:

$$\begin{aligned} p'_{ik} &= p_{ik} + \alpha.(e_{ij}.q_{kj} - \lambda.p_{ik}) \\ q'_{kj} &= q_{kj} + \alpha.(e_{ij}.p_{ik} - \lambda.q_{kj}) \end{aligned} \tag{5}$$

We implemented the above mentioned regularized MF with gradient descent on our restaurant reviews dataset. For our dataset a rank 20 approximation with the regularization parameter λ set to 0.2 gave us the lowest RMSE errors. Table 4 shows the errors in predicting the ratings on the three test sets. We observe that matrix factorization does not yield better results than our restaurant average-based strategy (Section 3.3). Our dataset is very sparse and a large number of rows and columns in the ratings matrix have almost all zero entries. Previous studies [3, 2, 32, 22] showed the usefulness of MF on the Netflix Challenge data [4], which has 40 times more known ratings in the training set as compared to our corpus. From the results in Table 4, we see that MF does not perform well in very sparse scenarios. Note that matrix factorization captures

		TestA	TestB	TestC
Predicting Star Rating	Using Star rating	1.130	1.259	1.124
	Using Textual rating	1.125	1.224	1.048
Predicting Textual rating	Using Star rating	0.704	0.719	0.767
	Using Textual rating	0.543	0.559	0.514

Table 5: Prediction RMSE using KNN for personalized predictions based on ratings.

both user and restaurant biases, and should more fairly be compared with the combined averages method of Section 3.3. In comparison to this baseline strategy, for the general Test Sets A and B the personalized prediction using MF performs marginally better.

Latent factor models have been successfully used in several previous studies [3, 2, 32, 35, 22]. However, MF does not yield sufficiently low errors on our sparse restaurant reviews corpus. In addition, latent factor models have low explainability; the meaning of the discovered latent factors is unclear. In the following section we experiment with neighborhood-based methods by grouping users based on the similarities in their rating behaviors.

4.1.2. Neighborhood Model

Our dataset has many more reviews for each restaurant on average than the average number of reviews per user. As a result, a restaurant average-based strategy performs well on our corpus as shown in Section 3.3. Therefore, we now focus on grouping similar users and make the prediction as the weighted average of the ratings given to the test restaurant by close neighbors.

We consider a K-Nearest Neighbor algorithm (KNN), a popular collaborative filtering technique [15], to identify the closest neighbors to a test user. After empirically comparing several distance functions, we computed the neighbors using a Pearson distance function with threshold [28] (our implementation uses a threshold value of 5). The threshold accounts for the number of items in common between users so that users are not considered as very close neighbors on the basis of only one common restaurant rated similarly.

The prediction algorithm uses the average of the K closest neighbors' scores (star rating or text rating) for the target restaurant as the predicted score. If a neighbor has not reviewed the restaurant, it uses the restaurant average-case prediction (Section 3.3) for that user.

We experimentally observed that the closest predictions were made when a close neighborhood of three users was used ($k = 3$). The resulting RMSE values are given in Table 5. The results are comparable to the baseline restaurant average-based prediction of Section 3.3; using close neighbors based on star or textual rating information does not help in improving rating predictions. In our sparse data users tend to review few restaurants making it difficult to find good neighbors that have reviewed the same restaurants and given similar ratings (cold start problem).

Using only the coarse ratings (star ratings or textually-derived ratings) for clustering is very restrictive. While the text-based ratings are derived using our sentence classification, all the information is combined into a single rating, making it difficult to distinguish the individual topics and sentiments covered in the review. Therefore, there is a need to use the full detailed textual information for finding like-minded users to make better personalized predictions, as described in the next section.

4.2. Text-based Personalized Prediction

We now explore enhanced techniques for finding similar users via clustering that utilize the textual information gained from the topical and sentiment classification. Unlike a hard clustering of users that assigns each user to exactly one cluster, soft clustering techniques assign users to every cluster with a probability greater than or equal to 0, and the sum of cluster membership probabilities for a given user equals to 1. There is evidence in the literature that in comparison to hard clustering, soft clustering is more robust to noise and performs better when the data cannot be separated into distinct clusters [10, 23]. Textual data is often fuzzy and a recommendation system built on such data will benefit from using probabilistic techniques for smoothening misclassification errors. Soft clustering captures the uncertainty in assigning values to clusters due to the similarity of values [8]. It also allows users to belong to different clusters with various degrees of confidence, allowing to represent for instance, user taste for both fine French cuisine and cheap Chinese dim sum. Therefore, we choose to implement a soft-clustering of the users to find similar users.

We use the Information Bottleneck (IB) Method [30] that assigns a probability to each user to belong to every cluster. The IB principle is described briefly in Section 4.2.1. In Section 4.2.2, we describe our adaptation of the iterative information bottleneck (iIB) algorithm [30] for clustering. We describe our novel prediction strategy using the cluster membership probabilities of users gained from the iIB algorithm in Section 4.2.3. The effects of parameter selections for the iIB method on the accuracy of ratings predictions are described in Section 4.2.4. Finally, after laying the groundwork, we describe experiments using the textual information in reviews as features for clustering in Section 4.2.5 and compare the prediction accuracy with the baseline restaurant average strategy of Section 3.3.

	R_1	R_2	R_3
U_1	4	-	-
U_2	2	5	4
U_3	4	*	3
U_4	5	2	-
U_5	-	-	1

Table 6: Ratings given by 5 users to 3 restaurants.

	R_1				R_2				R_3			
	Food Pos	Food Neg	Price Pos	Price Neg	Food Pos	Food Neg	Price Pos	Price Neg	Food Pos	Food Neg	Price Pos	Price Neg
U_1	0.6	0.2	0.2	-	-	-	-	-	-	-	-	-
U_2	0.3	0.6	0.1	-	0.9	-	0.1	-	0.6	0.1	0.2	0.1
U_3	0.7	0.1	0.15	0.05	-	-	-	-	0.2	0.8	-	-
U_4	0.9	0.05	0.05	-	0.3	0.4	0.2	0.1	-	-	-	-
U_5	-	-	-	-	-	-	-	-	-	0.7	0.3	-

Table 7: Matrix with four features as input to iIB algorithm.

	c_1	c_2	c_3
U_1	0.04	0.057	0.903
U_2	0.396	0.202	0.402
U_3	0.38	0.502	0.118
U_4	0.576	0.015	0.409
U_5	0.006	0.99	0.004

Table 8: iIB generated cluster membership probabilities.

4.2.1. Information Theoretic Clustering

The Information Bottleneck (IB) method was first introduced in [33] as an information-theoretic approach for data analysis and clustering. This method has been successfully used in document classification [31], unsupervised image clustering [14] and many other applications. We use the IB method in a collaborative filtering scenario to find similarity between users. The main principle behind the Information Bottleneck clustering is that the data is clustered or compressed such that the new compressed representation of the data retains the maximum possible amount of relevant information present in the data.

Let X be a discrete random variable distributed according to $p(x)$; the variable X represents the objects to be clustered. X contains information about another variable: the relevant variable Y. The goal of any clustering method is to cluster the data points in X such that the resulting clusters maintain most relevant information about Y. Let T, another random variable, denote the compressed or clustered representation of X. A soft clustering, as achieved using the IB method, is defined though a probabilistic mapping of each value $x \in X$ to each value $t \in T$. Therefore, the final output of the IB method is the membership probabilities of the data points in X in each of the clusters T.

The IB principle has its roots in Rate Distortion Theory. There can be several possible clusterings of the input variable X into the new representation T. One goal of clustering is to compress X, or to represent the input data points using a small number of clusters. Thus, the quality of the new representation T can be measured by its compactness. However, the compression is not enough. The *compression measure* can always be improved by ignoring details in X (e.g., by grouping all users in a single cluster), which will imply that the new representation T loses all relevant information about Y. Therefore, an additional constraint is needed; a *distortion measure* which represents the distance between the random variable X and its new representation T. The trade-off between the compactness of the new representation and its expected distortion is the fundamental trade-off in rate distortion theory.

Using the compression-distortion trade-off, the IB method aims to minimize the mutual information between X and its compressed representation T *(compression measure)*, under some constraint on the minimum mutual information that T preserves about the relevant variable Y *(distortion measure)*. In this sense, one is trying to squeeze the information X provides about Y through the compact “bottleneck” formed by the compressed representation T [30].

The trade off between the compression and distortion is parameterized by a single Lagrange parameter β. A large value of β ($\beta \to \infty$) indicates that the focus of the clustering is on the relevance of the underlying data, and the compression achieved through clustering is immaterial. In this case, each data point is put in a separate cluster of its own. On the contrary, a small value of β ($\beta \to 0$) assigns all data points in the same cluster, achieving maximum compression.

A detailed explanation of the IB method and the various algorithmic implementations can be found in [30]. In particular, we adapted the iterative information bottleneck (iIB) algorithm, and describe our implementation for the restaurant reviews data set in the following section.

4.2.2. Iterative Optimization Algorithm

We used the Iterative Information Bottleneck (iIB) algorithm, introduced in [33], to cluster like-minded users based on their reviewing behavior. As mentioned earlier, the goal is to cluster the input variable X via a probabilistic mapping to the variable T; while ensuring that T maintains maximum possible information about Y. Thus, in our case, the variable X represents the 30K users in our corpus. We use the different sentence types obtained from the text classification, represented as *(topic, sentiment)* pairs, as features for clustering. Therefore, the relevant variable Y represents the user preferences modeled by the information conveyed in the text of the reviews.

Consider the following artificial example consisting of a corpus of five users and three restaurants. The matrix representing the ratings given by these users to the restaurants is shown in Table 6; a blank matrix entry m_{ij} indicates that the corpus contains no review by user U_i for the restaurant R_j. Also, the $*$ in the m_{32} cell of the matrix indicates that we wish to predict the rating given by U_3 for R_2.

For simplicity, suppose that we cluster the five users based on only four sentence types; positive and negative sentences belonging to the food and the price categories (in actual experiments, all combinations of the six topics and four sentiment classes are used). This textually derived information is represented in a matrix shown in Table 7. The matrix shows that in the review written by U_1 for R_1 with 5 sentences, 3 were positive sentences about food, 1 was a food-related negative sentence and there was 1 positive price-related sentence. The entries in the matrix for each of the features is the normalized number of sentences of the feature type. (In actual experiments, the input matrix $P(X,Y)$ is a joint probability matrix, which is obtained from the matrix of restaurant-wise sentences of each type written by the users, similar to Table 7, after first ensuring that the sum of all entries in each row is 1, and then normalizing such that the sum of all entries in the matrix is 1.)

Given the input joint probabilities $P(X,Y)$, the iIB algorithm starts with a random initialization of the cluster membership probabilities $p(t|x)$. It then iteratively updates the probability matrix and converges to stable probability estimates [30]. The resulting output of the algorithm at the end of n iterations is a matrix $p^n(t|x)$ containing the *membership probabilities* of each user for each cluster.

Now, suppose we wish to cluster the users in Table 6 into 3 soft clusters. For our example the output matrix is shown in Table 8. As expected U_2 and U_3 are somewhat similarly clustered, while the clustering of U_1 or U_5 is distinct from all other users. These membership probabilities are then used for making personalized rating predictions (Section 4.2.3).

We experimented with several values for the cluster cardinality M and the trade-off parameter β. We use a sufficiently large value for the cluster cardinality

($M = 300$) and set $\beta = 20$. A brief comparison of the effects of parameter selection on prediction accuracy is outlined in Section 4.2.4.

4.2.3. Personalized Prediction Strategy

We now describe our novel rating prediction strategy based on a soft clustering of users. The output of the iIB algorithm is a soft clustering of the users X into T clusters with the probabilities given in $P^{(n)}(t|x)$, similar to Table 8. We use these probabilities to find the weights to be associated with the users who have reviewed the restaurant of interest, i.e., the restaurant in the test case. The predicted rating for a test case is the weighted average of the ratings of all other users who have reviewed the restaurant.

The weights model the similarities between users. Users who have similar cluster membership probabilities across all clusters are close neighbors. For each cluster, we first compute the cluster contribution as the weighted average of the ratings of all users who have reviewed the test restaurant. Formally, suppose we want to predict the rating given by the test user U_t to the test restaurant R_t. Let $Pr(U_t, R_t)$ denote this prediction. Assume that n users have reviewed the test restaurant with ratings: $rating(U_1, R_t)$, $rating(U_2, R_t)$, …, $rating(U_n, R_t)$. Also, for each user, U_1, U_2, …, U_n who has reviewed the test restaurant, let $U_1(c_i)$, $U_2(c_i)$, …, $U_n(c_i)$ denote the probabilities with which these users belong to a cluster c_i. Now, the contribution for a cluster c_i is given by:

$$Contribution(c_i, R_t) = \frac{\sum_{j=1}^{n} U_j(c_i) * rating(U_j, R_t)}{\sum_{j=1}^{n} U_j(c_i)} \tag{6}$$

Furthermore, we have M clusters, say c_1, c_2, …, c_m. The final prediction for the test review takes into account the cluster membership probabilities of the test user $U_t(c_i)$ to compute a weighted sum of the individual cluster contributions from Equation 6. Therefore, the final prediction $Pr(U_t, R_t)$ is given by the following formula:

$$Pr(U_t, R_t) = \frac{\sum_{i=1}^{m} U_t(c_i) * Contribution(c_i, R_t)}{\sum_{i=1}^{m} U_t(c_i)} \tag{7}$$

Consider the example in Section 4.2.2 again. Suppose we want to predict the rating given by U_3 to R_2. There are two other users (U_2 and U_4), who have reviewed this restaurant. For each of our three clusters, we find the cluster contribution as the weighted sum of the ratings given by these two users to the test restaurant R_2. Using Equation 6, and the matrices in Table 6 and Table 8, we have:

$$\begin{aligned} Contribution(c_1, R_2) &= \frac{\sum_{j=2,4} U_j(c_1) * rating(U_j, R_2)}{\sum_{j=2,4} U_j(c_1)} \\ &= \frac{0.396 * 5 + 0.576 * 2}{0.396 + 0.576} = 3.222 \end{aligned}$$

Similarly, for the other clusters; $Contribution(c_2, R_2) = 4.793$ and $Contribution(c_3, R_2) = 3.487$. The final prediction for User U_3 and Restaurant R_2 is computed using

Equation 7 and the cluster membership probabilities of the test user (U_3) from Table 8; given by:

$$\begin{aligned} Pr(U_3, R_2) &= \frac{\sum_{i=1}^{3} U_3(c_i) * Contribution(c_i, R_2)}{\sum_{i=1}^{3} U_3(c_i)} \\ &= \frac{0.38 * 3.222 + 0.502 * 4.793 + 0.118 * 3.487}{0.38 + 0.502 + 0.118} = 4.04 \end{aligned}$$

This predicted value is compared with the actual rating given by the user, to compute the error in prediction.

4.2.4. Parameter Selection

The two input parameters to the iIB algorithm are the cluster cardinality parameter M, and the Lagrange parameter β that determines the trade-off between the compression and the relevance of a clustering. The parameter M needs to be large enough for the data points to be clustered. However, the complexity of the algorithm increases linearly with an increase in the number of clusters. Although, it is possible to run the algorithm offline with periodic updates or to speed up the computation using distributed processing; in our experiments we observed diminishing and unclear improvements in prediction accuracy as the number of clusters increased above $M = 300$. Therefore, for the iIB experiments we fix the number of clusters to 300.

The selection of the trade-off parameter β is more interesting as the prediction accuracy clearly differs with different values for this parameter. For low values of β, implying that the primary focus of the clustering is on the compression of the data, all users are clustered similarly. This makes the weighted restaurant average of the iIB algorithm very similar to the baseline restaurant average of Section 3.3. Figure 1 shows the percentage improvement of the accuracy of the iIB method using textual features over the accuracy of the restaurant average prediction of Section 3.3, for different values of β. The figure represents the accuracy for the task of predicting the star ratings for our three experimental test sets (Section 2.3) as β increases from 1 to 30 (with $M = 300$ clusters). We notice that, initially as β increases, there is a steady improvement in prediction accuracy. However, after $\beta = 20$ there is an increase in the error. This can be explained by the fact that as β increases to very high values, the compression achieved via clustering become irrelevant. This results in poor grouping of users, in turn causing the error values to increase. The clustering for our techniques is done offline and the actual overhead is transparent to the users. All clustering-based recommendation algorithms require this offline step, and it does not impact the actual recommendation time from a user's perspective. An open research direction is to adapt our algorithm to handle incremental data updates without recomputing the entire clustering; this is left for future work.

4.2.5. Clustering based on Full Textual Features

We first experimented with using the iIB method with only the star-ratings matrix converted to the input joint probability $P(X, Y)$. However, as expected

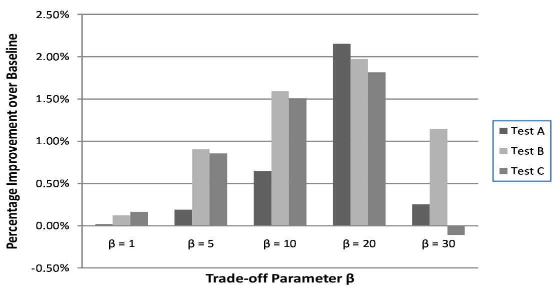


Figure 1: Effect of varying β on prediction accuracy.

the improvements in prediction accuracy over the corresponding results obtained via the rating-based methods (Section 4.1), were marginal. The star ratings lack in conveying all the rich information present in the text of the reviews: a user should not be suggested a restaurant, where the overall rating is reflective of topics in which a user is not interested, as illustrated in Example 1. By using the topics and sentiments present in user reviews, we can derive user groupings that take into account the individual interests of users. Therefore, we use the textual information obtained from our classification for clustering users. This allows us to cluster users not only based on the commonality of the restaurants reviewed, but also on their text reviewing patterns or habits.

For the textual content experiments, the input matrix $P(X,Y)$ contains features representing the different sentence types in the text of the review, for each of the 5531 restaurants. In this case, our features mirror the reviewing behaviors of the users, represented by the topics of sentences in the review and the sentiments towards these topics. For the experiment in this section, we used the full textual information derived from the reviews. Therefore, for each restaurant in the data set, we have 34 sentence types representing all combinations of the sentence topics and sentiments (sentences can have a combination of a topic and a sentiment, or one of either), resulting in about 190K features.

Table 9 shows the RMSE errors in the predictions for the three test sets when the richer textual information is used as features for the iIB clustering. Note that in all cases, the clustering is done using sentence features, but different ratings (star or text) are used for making predictions. Using textual information for personalized prediction always yields lower error values than the rating-based personalized predictions of Section 4.1.2 (Table 5) and the matrix factorization method of Section 4.1.1 (Table 5). Moreover, in comparison to the restaurant average-based predictions of Section 3.3 (Table 3), the improvements in RMSE values shown in the results presented in Table 9 are statistically significant ($p - value < 0.05$ using the one-sided Wilcoxon test) for all test sets for the task of predicting unknown star ratings using training data star ratings; for the task of predicting star ratings using training data text ratings, the

		TestA	TestB	TestC
Predicting Star Rating	Using Star rating	1.103	1.242	1.106
	Using Textual rating	1.113	1.211	1.046
Predicting Textual rating	Using Star rating	0.692	0.704	0.742
	Using Textual rating	0.544	0.549	0.514

Table 9: Prediction RMSE using full textual content for personalized predictions.

improvements in RMSE values shown in the results presented in Table 9 are statistically significant over those of Table 3 for the randomly chosen Test sets A and B.

Comparing the personalized predictions based on using coarse rating information (Section 4.1) and on using the review text content (Table 9) for grouping users, we see that for the traditional recommendation task of predicting unknown star ratings using the training data star ratings, our three test sets A, B and C show a 2.41%, 1.34% and a 1.65% (resp.) improvements when textual information is used.

An important task for a recommendation system is to return the best k product choices for a user. In [21], the author shows that a small improvement in RMSE (even as low as 1%) has a significant impact on the precision of top-k lists. Achieving improvements in prediction accuracy is a notably hard task. The recent Netflix challenge [4], awarded a prize of 1 million dollars to a team achieving a 10% improvement over the existing algorithm; along with step prizes for each 1% improvement. This shows that our methods of incorporating review text in a recommendation system have significant benefits for collaborative filtering systems.

In conclusion, the error values in Table 9 show that using the textual information in conjunction with the iIB clustering algorithm improves on the baseline restaurant-average prediction from Table 3. Moreover, for our dataset this method is more adept at making personalized prediction than the KNN-based predictions of Section 4.1.2 and the factorization-based method of Section 4.1.1. Thus, our techniques demonstrate that the largely untapped textual information in user reviews contains very rich and detailed information that can be effectively used in a text-based recommendation system to improve rating predictions.

5. Qualitative Prediction of Review Components

An important task in understanding and analyzing user reviews is the ability to make fine-grained predictions on the actual content in the reviews. Several websites like TripAdvisor and Yelp have recognized the need for presenting a summary of sentiment towards different product features. Some web sites such as Citysearch, provide binary yes-no answers to questions pertaining to the Ambience and the Service of each restaurant (Romantic? Prompt Seating? Good for Groups?) as well as a numeric Price level. However, this limited summary information is gathered by asking reviewers several yes-or-no questions, making the task of writing reviews very daunting. In addition, the information presented to users is not personalized to match their tastes.

In this section, we describe our techniques for making fine-grained predictions of user sentiments towards the different restaurant aspects, derived automatically from the text of the reviews. First, we cluster users based on their opinions about an individual aspect of the restaurant (Section 5.1); such specialized clustering results in neighbors who have the same sentiment towards the particular restaurant aspect. We use the cluster membership probabilities derived from this topic-wise clustering, to predict the importance that a user will assign for each feature and sentiment in his review. We then translate these predictions to binary like/dislike judgments (Section 5.2) towards each restaurant feature and evaluate the prediction accuracy in Section 5.3.

5.1. ***Clustering based on Restaurant Topics***

The average and personalized predictions of Sections 3 and Section 4 provide an overall predicted rating for a restaurant that does not differentiate on the various restaurant features. Yet, a user might have different sentiments towards a given restaurant: for instance liking the Food and Price but disliking the Service. To accurately predict the sentiment of the user towards each *individual* aspect of the restaurant (Food, Service, Price, Ambience, Anecdotes, and Miscellaneous), we cluster users along six dimensions, using the sentences belonging to each of the six restaurant topics separately. For each user we obtain six sets of neighbors, one for each identified topic.

We cluster users using the information bottleneck method described in Section 4.2 with the features belonging to a particular topic. For each restaurant in the data set, we use 5 sentence types features representing all combinations of the sentiments for a particular topic (sentences belonging to a topic can have one of the four sentiments: Positive, Negative, Neutral, or Conflict, or no sentiment), for clustering. The resulting cluster membership probabilities indicate the topic-wise similarity between users; users who have similar sentiment towards the topic across all commonly reviewed restaurants are clustered together. Using the cluster membership probabilities, we now predict the percentage of sentences belonging to each sentence type; not a numeric rating as discussed in Section 4.2. The sentence proportion belonging to a particular *(topic, sentiment)* pair is the weighted average of the proportion of sentences of that type written in the other reviews of the restaurant. This weighted average is computed using the prediction algorithm described in Section 4.2.3, where the neighbor ratings are replaced by their sentence type proportions. Therefore, we have predictions for the proportion of sentences of each type that a user may write for the restaurant. In the following section, we describe how these predicted sentence proportions are translated into qualitative binary like/dislike predictions.

5.2. ***Topical Sentiment Prediction***

We are interested in determining qualitatively whether a user will like (is predicted to have a positive sentiment towards) or dislike (is predicted to have a negative sentiment towards) a particular restaurant aspect. Our data does

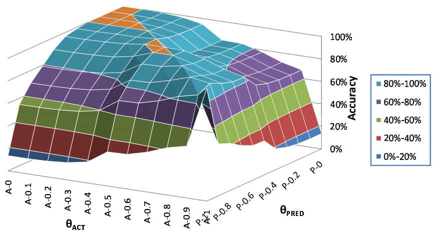


Figure 2: Prediction accuracy for positive ambience reviews with varying threshold values.

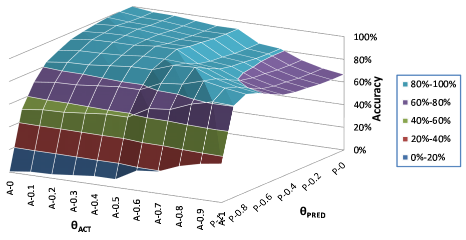


Figure 3: Prediction accuracy for negative ambience reviews with varying threshold values.

not contain any ground truth either in the form of binary judgments or ratings for the user sentiment towards the individual restaurant aspects. Therefore, we make sentiment predictions using the predicted proportion of positive and negative sentences belonging to a particular topic (Section 5.2.1). We next learn the optimal parameters for making highly accurate sentiment predictions in Section 5.2.2 and evaluate our predictions accuracy and F1-score.

5.2.1. Parameters for Sentiment Prediction

For each restaurant topic, we need to determine two thresholds: θ_{pred} and θ_{act}. For a topic, if our predicted review composition contains a proportion of positive sentences greater than θ_{pred}, we predict that the user will like this restaurant aspect. Similarly, if our prediction contains a proportion of negative sentences greater than or equal to $(1-\theta_{pred})$, we predict that the user will dislike the restaurant aspect. Reviews which do not meet either of the conditions above (due to the existence of Neutral and Conflict sentiment sentences) are predicted to be neutral reviews; for such reviews we cannot make polar judgment predictions.

To evaluate our predictions we also need to determine whether the actual review (in the test set) indicates that the user will like or dislike the particular restaurant aspect. Therefore, for each restaurant topic we define a threshold for the actual user judgment: θ_{act}. The actual judgment towards a restaurant aspect is considered to be positive if the review contains a proportion of posi-

	θ_{act}	θ_{pred}	Combined Accuracy	Positive F1	Negative F1
Food	0.5	0.5	73%	0.85	0.19
Price	0.5	0.5	76%	0.86	0.49
Service	0.5	0.5	61%	0.76	0.22
Ambience	0.5	0.5	76%	0.86	0.49
Anecdotes	0.5	0.5	65%	0.78	0.36
Miscellaneous	0.5	0.5	63%	0.77	0.25

Table 10: Evaluating sentiment predictions with ($\theta_{act} = 0.5, \theta_{pred} = 0.5$).

tive sentences greater than θ_{act}, if the review contains a proportion of negative sentences greater than or equal to $(1 - \theta_{act})$ the review is considered to be negative, else the review is considered to be actually neutral towards the particular restaurant aspect.

5.2.2. Learning from the Data

We created topic-wise development sets of 215 reviews for each restaurant topic, to empirically determine the threshold values for each topic. Using the training sets, we predicted the review composition for each review set aside in the development sets. We use accuracy as the metric for evaluating our predictions. Accuracy measures the proportion of correctly predicted reviews (true positives and true negatives) to the total number of predictions.

For each restaurant topic, we varied both the actual and predicted parameters. Figure 2 shows the accuracy for predicting whether a user will like the ambience in a restaurant. We see that at ($\theta_{act} = 0, \theta_{pred} = 0$), we predict all reviews to be positive about the ambience and trivially achieve an accuracy of 100%. Fixing $\theta_{act} = 0$, as we increase the prediction threshold θ_{pred}, we predict fewer reviews to be positive on the ambience and the accuracy gradually decreases (true positives decrease and false negatives increase). Similarly fixing $\theta_{pred} = 0$, as we increase the actual threshold θ_{act} the accuracy decreases as true negatives decrease and false positives increase. Interestingly, we get a high prediction accuracy of 95% when we set ($\theta_{act} = 0.8, \theta_{pred} = 0.8$). This implies that even though we are quite selective in assuming that the review is a positive ambience related review, our prediction methods are able to capture the sentiment with a very high accuracy.

For predicting whether a user will dislike the ambience in a restaurant, the accuracy at varying thresholds is shown in Figure 3. Again, we achieve a good accuracy (93%) when we set ($\theta_{act} = 0.8, \theta_{pred} = 0.8$), as described above. The threshold values set at 0.8 indicate that for a review to be deemed positive on ambience it needs to have more than 80% positive ambience related sentences; whereas if the negative sentences occur only 20% times, the review is deemed negative. This is consistent with our observations of the sentiment distribution [13]. We have similar trends with varying thresholds for the other 5 restaurant topics, and omit the accuracy plots due to space limitations.

We next evaluate the review sentiment prediction using combined accuracy and F1 scores. The combined accuracy is computed as the proportion of all correct predictions (positive, negative or neutral) to the total number of reviews. Unlike the separate assessment of positive and negative accuracy in the plots

	θ_{act}	θ_{pred}	Combined Accuracy	Positive F1	Negative F1
Food	0.8	0.8	78%	0.87	0.80
Price	0.7	0.7	86%	0.91	0.92
Service	0.7	0.7	70%	0.80	0.72
Ambience	0.8	0.8	85%	0.89	0.87
Anecdotes	0.6	0.6	69%	0.81	0.58
Miscellaneous	0.8	0.8	67%	0.81	0.63

Table 11: Evaluating sentiment predictions with threshold parameters learned from the text.

above, the combined accuracy is more strict as it does not benefit much from many true negatives. We set the thresholds as $(\theta_{act} = 0.5, \theta_{pred} = 0.5)$, and show the prediction accuracies in Table 10. We achieve a good combined accuracy (>73%) for only the Food, Price and Ambience categories. We also include the F1-scores for predicting the positive and negative sentiments. Our results show that we achieve high F1-scores (>76%) for making positive sentiment predictions for all topics, but very low F1-scores for negative predictions.

Fixing the threshold parameters to $(\theta_{act} = 0.5, \theta_{pred} = 0.5)$ is not representative for our corpus. Due to the skew towards positive sentiment in our corpus, for a review to be considered positive on a topic the threshold parameters should be higher than 0.5. Table 11 shows the accuracy and F1 scores when the threshold parameters mirror the distribution of positive and negative sentiment towards each topic in our data. A threshold value of 0.8 for the Food category indicates that the positive food related sentences and negative food related sentences have a 80-20 distribution in our classified data. As seen in Table 11, learning the threshold values from the text itself, results in a high combined accuracy (>70%) for the main categories of Food, Price, Service and Ambience. Anecdotes and Miscellaneous topics yield lower accuracy values. However, qualitative judgment predictions for these topics do not add much to the user experience. Note that with threshold values learned from the sentiment distribution, we achieve very high F1 scores for both the positive and the negative sentiments, unlike the results in Table 10. A high F1 score for the negative sentiment indicates that our techniques are proficient in detecting the negative judgement in the reviews with high precision and recall; a task that is notably hard due to the lack of sufficient negative examples. Therefore, we set the threshold parameters for the different topics to the values in Table 11.

In the following section we discuss an example system that utilizes our text-based rating prediction from Section 4.2, as well as the qualitative binary predictions.

5.3. ***Example Interface and Evaluation***

Our methods allow us to make rating predictions which indicate the general user assessment of the restaurant, as well as fine-grained qualitative predictions about user sentiment towards individual restaurant features. The key point is that these predictions are made automatically by deriving useful information from the textual content in reviews.

Figure 4 shows an example interface for a system built using our techniques. As shown, a user can search for a restaurant and we provide text-based predictions. To evaluate the quantitative rating predictions and the qualitative judgment predictions of such a system, we set aside a new joint-predictions test set containing 30 reviews, and where each user has reviewed at least five restaurants. For the new test set our text-based methods from Section 4.2.5 (text for clustering, as well as textual ratings for predictions) result in a RMSE value of 1.043. For the same 30 test reviews a star rating-based neighborhood model (Section 4.1.2) results in a RMSE error of 1.210. Hence, our text-based techniques show a 13.8% improvement over the star rating-based system. In addition, for this new test set we provide sentiment prediction for the individual restaurant features, using the threshold parameters shown in Table 11. The sentiment predictions have a combined accuracy of 81.8%; indicating that our techniques are proficient in capturing the information present in the review text to make fine-grained personalized predictions.

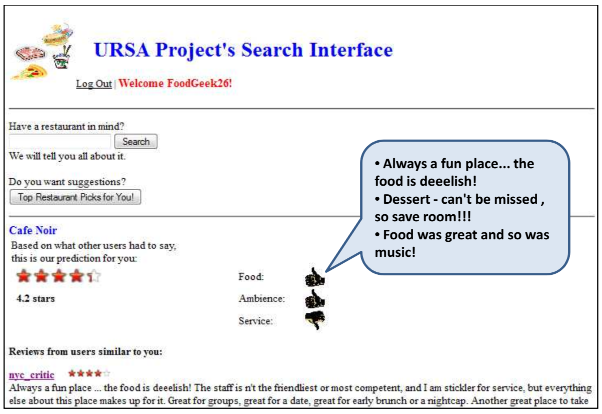
Our interface also offers an alternate way to accessing the information present in the textual reviews, by providing example sentences belonging to the different *(topic,sentiment)* types. Therefore, a user no longer has to browse through the large amount of unstructured text in the reviews, but can browse a few sentences for each topic and sentiment that reflect the characteristics of the restaurant. Our future work includes choosing the best sentences to be displayed to the user based on length, number of nouns and adjectives, frequently repeating phrases, and other indicators.

Our novel qualitative predictions of individual features is a promising direction to follow to understand and analyze user reviews in detail.

6. Related Work

Online reviews are a useful resource for tapping into the vibe of the customers. Accessing and searching text reviews, however, is often frustrating when users only have a vague idea of the product or its features and they need a recommendation. The design of a good recommender system has been the focus of many previous work; a good survey of the work done in this area and the comparison of several techniques is found in [15] and [5]. Recently, the Netflix challenge [4] has brought a lot of attention to collaborative filtering and recommendation systems. The Netflix data as well as the data typically used in other projects on recommendation systems like the pioneer GroupLens project [27], consists of highly structured metadata, often only the rating given by a user to a product. In contrast, our work considers the textual content of reviews to make predictions.

Identifying both topical and sentiment information in the text of a review is an open research question. Review processing has focused on identifying sentiments, product features [9] or a combination of both [16, 1, 34]. An alternate approach to identifying textual features and sentiments expressed towards them is to use unsupervised classification which has the advantage of not requiring a human-annotated set for training classifiers. In [6], the authors present a unsupervised text classification technique for the Citysearch restaurant reviews data set used in this paper.


URSA Project's Search Interface

Log Out | Welcome FoodGeek26!

Have a restaurant in mind?

Search

We will tell you all about it.

Do you want suggestions?

Top Restaurant Picks for You!

Cafe Noir

Based on what other users had to say, this is our prediction for you:

4.2 stars

Food:

Ambience:

Service:

• Always a fun place... the food is deeelish!
• Dessert - can't be missed , so save room!!!
• Food was great and so was music!

Reviews from users similar to you:

nyc_critic

Always a fun place ... the food is deeelish! The staff is n't the friendliest or most competent, and I am stickler for service, but everything else about this place makes up for it. Great for groups, great for a date, great for early brunch or a nightcap. Another great place to take

Figure 4: Example search interface with rating predictions and fine-grained sentiment predictions.

Studies like [16] focus on identifying individual product features and sentiments. However, unlike our work these studies do not use the extracted opinions and features for collaborative filtering. Most of the work in sentiment analysis operates at the review level. Our processing unit is a sentence, so that a review is modeled as a fine-grained combination of topics and sentiments.

With the advent of online user generated content, social networks and online shopping, recommendation systems have seen a surge in popularity. The recent work by Wang and Blei [35], uses matrix factorization for making predictions for previously rated items as well as items that have never been rated (cold start). Similar to our work, the authors use topic modeling to capture user preferences. In [11], the authors enhance a matrix factorization-based recommendation system by mapping user or item attributes to the latent factors to make predictions for new users or new items. In [22], the winners of the popular Netflix Challenge demonstrate the effectiveness of matrix factorization techniques in making accurate recommendations, and claim that latent factor models often outperform neighborhood based models. However, our results in Section 4.1.1 show that matrix factorization does not reduce prediction errors for our sparse dataset. In fact, several recent studies like [17] demonstrate the effectiveness of an ensemble or a blend of several individual techniques, and show that ensemble-based methods outperform any single algorithm. Our soft clustering-based models can be used effectively in such ensembles, and wish to explore this in the future.

The recent work by Leung, Chan and Chung [19] incorporates review text analysis in a collaborative filtering system. While the authors identify features, they unfortunately do not describe their methods and do not summarize all their features or roles. Additionally, the evaluation of their recommendation is done by predicting a 2-point or a 3-point rating. We predict ratings at a fine-grained 5-point rating scale, commonly used in popular online reviewing systems.

The approach in [25] identifies aspects or topics by clustering phrases in textual comments, and identifies user sentiment towards these aspects. However, their techniques often result in finding noisy aspects. In addition, the aspect clustering precision and recall (0.59, 0.64) for their experiments is lower than the average topic classification precision and recall (0.70, 0.64) for our sentence topical classification (Table 1). The study in [25] makes a aspect rating prediction and combines these ratings to make a prediction on the overall rating in the review. However, the predictions do not utilize the ratings of similar users to the product, therefore ignoring the social impact of other users on the user assessment of a product. Our soft clustering method groups users according to their similarities of reviewing behavior, and hence captures the underlying inter-dependencies between user ratings.

7. Conclusions and Future Work

In this paper, we presented the user reviews classification and analysis effort performed as part of our URSA project. Our main contribution is the assessment of the impact of text-derived information in a recommendation system. We show that both topic and sentiment information at the sentence level are useful information to leverage in a review. In addition, we use soft clustering techniques to group like-minded users for personalized recommendations, using the *detailed textual structure and sentiment of reviews.* Our techniques make better ratings predictions using the textual data, and moreover, we make fine-grained predictions of user sentiment towards individual restaurant features.

We are investigating additional refinements to our text-based recommendations, including better text classification strategies and utilizing temporal factors and other available metadata to guide our analysis. In addition, we are interested in the impact of text classification on search over reviews and are implementing tools that allow users to search reviews using topic and sentiment information. Lastly, similar to the study in [7] we are interested in evaluating the performance of our techniques in generating top-k restaurant recommendation lists.

We make our data available at http://spidr-ursa.rutgers.edu/datasets and our code for making personalized predictions using the Information Bottleneck method at http://spidr-ursa.rutgers.edu/code.

Acknowledgment

This work was partially supported by a Google Research Award and NSF grant IIS-0844935.

References

[1] N. Archak, A. Ghose, and P. G. Ipeirotis. Show me the money!: deriving the pricing power of product features by mining consumer reviews. In *SIGKDD*, 2007.

[2] R. M. Bell and Y. Koren. Lessons from the netflix prize challenge. *SIGKDD Explorations Newsletter*, pages 75–79, December 2007.

[3] R. M. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *IEEE International Conference on Data Mining*, 2007.

[4] J. Bennett and S. Lanning. The netflix prize. In *KDD Cup and Workshop*, 2007.

[5] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Conference on Uncertainty of Artificial Intelligence*, pages 43–52, 1998.

[6] S. Brody and N. Elhadad. An unsupervised aspect-sentiment model for online reviews. In *HLT-NAACL Conference of the North American Chapter of the Association for Computational Linguistics*, 2010.

[7] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 39–46, 2010.

[8] B. T. Dai, N. Koudas, B. C. Ooi, D. Srivastava, and S. Venkatasubramanian. Rapid identification of column heterogeneity. *IEEE International Conference on Data Mining*, pages 159–170, 2006.

[9] K. Dave. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *WWW*, pages 519–528, 2003.

[10] M. Futschik and B. Carlisle. Noise-robust soft clustering of gene expression time-course data. In *Journal of Bioinformatics and Computational Biology*, 2005.

[11] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thie. Learning attribute-to-feature mappings for cold-start recommendations. ICDM '10, pages 176–185, 2010.

[12] G. Ganu, N. Elhadad, and A. Marian. Beyond the stars: Improving rating predictions using review text content. In *WebDB*, 2009.

[13] G. Ganu, A. Marian, and N. Elhadad. *URSA - User Review Structure Analysis: Understanding Online Reviewing Trends*, 2010. DCS Technical Report No. 668.

[14] J. Goldberger, H. Greenspan, and S. Gordon. Unsupervised image clustering using the information bottleneck method. In *DAGM Symposium on Pattern Recognition*, 2002.

[15] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, pages 5–53, 2004.

[16] M. Hu and B. Liu. Mining and summarizing customer reviews. In *SIGKDD*, pages 168–177, 2004.

[17] M. Jahrer, A. Töscher, and R. Legenstein. Combining predictions for accurate recommender systems. In *SIGKDD*, pages 693–702, 2010.

[18] T. Joachims. A support vector method for multivariate performance measures. In *International Conference on Machine Learning*, 2005.

[19] C. W. ki Leung, S. C. fai Chan, and F. lai Chung. Integrating collaborative filtering and sentiment analysis: A rating inference approach. In *ECAI-Workshop on Recommender Systems*, pages 62–66, 2006.

[20] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conferences on Artificial Intelligence*, 1995.

[21] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *SIGKDD*, pages 426–434, 2008.

[22] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42:30–37, August 2009.

[23] K.-I. Lin and R. Kondadadi. A similarity-based soft clustering algorithm for documents. In *DASFAA: International Conference on Database Systems for Advanced Applications*, pages 40–47, 2001.

[24] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, pages 76–80, 2003.

[25] Y. Lu, C. Zhai, and N. Sundaresan. Rated aspect summarization of short comments. In *WWW*, 2009.

[26] W. Mendenhall and T. L. Sincich. *A Second Course in Statistics: Regression Analysis (6th Edition)*. Prentice Hall.

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

[28] J. L. Rodgers and W. A. Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, pages 59–66, 1988.

[29] S. Siegel and J. N. John Castellan. *Nonparametric Statistics for the Behavioral Sciences, Second Edition*. McGraw-Hill, 1988.

[30] N. Slonim. *The Information Bottleneck: Theory and Applications*. PhD thesis, Hebrew University, 2002.

[31] N. Slonim, N. Friedman, and N. Tishby. Unsupervised document classification using sequential information maximization. In *SIGIR*, 2002.

[32] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Matrix factorization and neighbor based algorithms for the netflix prize problem. In *RecSys*, pages 267–274, 2008.

[33] N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method. In *Annual Allerton Conference on Communication, Control and Computing*, 1999.

[34] I. Titov and R. McDonald. A joint model of text and aspect ratings for sentiment summarization. In *ACL Annual Meeting of the Association of Computational Linguistics*, 2008.

[35] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *SIGKDD*, pages 448–456, 2011.