RYERSON UNIVERSITY

REACHING BEYOND THE STARS IN RECOMMENDING THAI RESTAURANTS IN LAS VEGAS: A SENTIMENT DETECTING APPROACH TO RATE REVIEWS AS A COMPLEMENT TO USER RATINGS

CAPSTONE PROJECT

PRESENTED TO

BORA ÇAGLAYAN

CKME 136

BY

LUC DUFF
SN500731705

DECEMBER 2015

Reaching beyond the stars in recommending Thai restaurants in Las Vegas: a sentiment detection approach to rate reviews as a complement to user ratings

ABSTRACT

In line with career prospects in social networking applications, this classification project deals with recommender systems and focuses on the problem of matching a user to a new item. Based on the business and review academic datasets from the 2015 Yelp Dataset Challenge, the task is to classify a Thai restaurant in Las Vegas that is new to a user as a restaurant for the user to experience or not. Performance is measured, as a matter of classification accuracy, by a restaurant rating that is predicted correctly for a user from user ratings and from lexicon-extracted sentiment scores in review text. The merged working dataset has 990,627 reviews, of which 405,760 target one of the 4,960 restaurants in Las Vegas. In line with the "Yelp Restaurant Lexicon" (Kiritchenko, Zhu, Cherry, & Mohammad, 2014b) whose units of analysis are reviews of Yelp Restaurants in Phoenix (AZ), we narrow down the reviews to reviewers of restaurants in Phoenix, although not exclusively. We further narrow the reviews to Thai restaurants to accommodate R and RAM limits. Using restaurant ratings given by reviewers and sentiment scores, we select a user-based collaborative approach to predict missing ratings and to deliver to the user top recommendations.

1. Introduction

While recommender systems have been around for more than a decade now, their poor performance on some of the social media websites and the excellent performance on the Google search engine make you wonder how it all works. Recommender systems are first classified into three major categories: content-based, collaborative and hybrid. While generally using ratings for recommendations, collaborative recommenders are divided into classes: user-based and item-based. User-based classification systems focus on user-to-user similarities in taste whereas item-based classification systems make the most of item-to-item similarities in ratings. In this project, we focus on user-to-user similarities and use review ratings and sentiment scores extracted from reviews with two lexicons –the Yelp Restaurant Lexicon (Kiritchenko, et al., 2014b) and a customized lexicon combining two AFINN lexicons (F. Nielsen, 2015; F. A. Nielsen, 2011)- to bring forward top recommendations of Thai restaurants in Las Vegas that are new to a user. We narrow this user to reviewers of restaurants in Phoenix (AZ) for purposes of compatibility with the

Yelp Restaurant Lexicon that was created from Yelp reviews of restaurants in Phoenix. We assume that the choice of words by users who rate Phoenix restaurants in the working dataset will remain consistent across reviews of restaurants in Phoenix and Las Vegas. We rely on the relatively short-distance between Phoenix and Las Vegas as a factor for maintaining this assumption: residents of Phoenix and travellers to Phoenix alike travel to Las Vegas. We also understand that Las Vegas residents may not be as compelled to travel to Phoenix; therefore, reviewers who are Las Vegas residents are likely under-represented in this sample.

In this project, we assume that sentiment scores are measures of the quality of the user's experience: a measure of satisfaction that can't ignore price (i.e.: quality-price ratio). For time constraint reasons, we choose not to break the restaurant experience into restaurant domain topics, although pricing is considered statistically. This report addresses the following topics: the literature review, the description of the data, the approach and the initial results. The final results, conclusion and discussion will be added at a later date for the final submission.

2. The Literature Review

a) Related works on recommender systems

Jannach, Zanker, Felfering and Friedrich (2011) argue that the purpose of the recommender system field is to construct systems that support the decision making of online users. In turn, recommender systems aim to provide easy and timely access to high-quality and meaningful recommendations to a large community of users with different tastes, backgrounds and preferences. Therefore, computation-heavy recommendation algorithms need to produce outputs before the user comes online while computation-light algorithms can be used as it happens in real time. In collaborative recommendation, Jannach et al. (2011) argue that information on past behavior and opinions (ideas, beliefs, sentiments, etc.) shared by a community of users can be used for the prediction of items the user will more likely be interested in. Ultimately, recommendations can be used by the social-network application to promote items beyond the user's normal selection and by application clients to increase sales.

Collaborative recommenders perform typically two tasks: (1) they predict the rating for an item that the user has not yet rated, and (2) they create a list of recommended items that is tailored to the user. (Bigdata Doc, 2014) A typical collaborative approach works with a matrix of user-item ratings given by the user as "the only input" (a bonus for faster computations) and produces predictions that reveal the degree of a user's interest in a specific item and a list of top recommended items. (Jannach, et al., 2011) The ratings matrix has single items laid out horizontally at the top, single users laid out vertically on the left, and ratings laid across the matrix, each representing the rating of a single user for a single item. (Hahsler, 2015b, p. 4) A rating that is missing from a cell in the matrix establishes that the user has not rated/seen/reviewed the item yet. Caraciolo (2013) argues that the main challenge in collaborative recommenders is to estimate the values that are missing in the matrix. Ultimately, once ratings have been predicted for items that had missing values, the objective is to select the items that have the highest predicted ratings and to present a list of recommendations from the top rating predictions to the user. (Hahsler, 2015b, p. 4) Navisro Analytics (2012), a full-service consulting firm in San Jose, CA, offers a more extended breakdown of the stages and steps in the delivery of recommendations. An adapted version of this breakdown serves as the foundation for the three-recommender exercises proposed in this project.

Lastly, Caraciolo (2013) and Ganu, Kakodkar and Marian (2012) warn that data sparcity, also known as the cold start problem, is typically a problem in recommender systems. Caraciolo (2013) argues that similarities between users and items are difficult to compute when dealing with a sparse matrix: a critical mass of user/item exposure is key to the success of a recommender action. In order to resolve the sparcity problem in the Yelp dataset, Carillo et al. (2013) substituted the missing rating for the user's personal average rating. (p. 5) A rating scheme of 1 (strongly dislike) to 5 (strongly like) is common in the field. (Caraciolo, 2013; Caraciolo, Melo, & Caspiro, 2011; Hahsler, 2015a; Jannach, et al., 2011; Yang, Zhang, Yu, & Zhu, 2013) The predicted ratings replacing the missing values in the matrix typically fall within this range. (Hahsler, 2015b)

b) Related works on recommender packages

In this project, we consider 2 recommender packages: recommenderlab for R and crab for Python. We select R in this project as recommenderlab has been kept up-to-date whereas crab for Python has been left adrift since 2011. As R demands much RAM for the deployment of a rating matrix, the recommender prototype in this project turns to Thai restaurant reviews in the Yelp dataset.

The package 'recommenderlab' for R was designed mainly for collaborative recommender problems (Hahsler, 2015b) while it was meant for development and testing purposes. (Hahsler, 2015a) Among other things, the package can (1) calculate the prediction error for a recommendation, (2) calculate the (dis)similarity between rating data by users and for items, (3) perform a simple split into training and test data, k-fold cross evaluation, k bookstrap samples, (4) (de)normalize ratings, (5) create precision-recall and ROC plots for evaluation results, (6) predict recommendations using data about new users, (7) create a rating matrix from given data, and (8) provide a list of top recommendations. (Hahsler, 2015a) Bigdata Doc (2014) provides a basic implementation framework and code using recommenderlab for a recommendation project with a non-identified dataset while Bhatnagar (2012) provides code for recommendations based on a MovieLense dataset. Both Bigdata Doc (2014) and Bhatnagar (2012) used RMSE as a measure of accuracy. Rennie (2015) and vakerkamachi (2015) provide code for a recommender project based on a Yelp dataset and on recommenderlab for R.

c) Related works on sentiment analysis

There are three major types of approaches to sentiment analysis: (1) the lexicon-based approach (lexicon dependent), (2) the machine learning-based approaches, including supervised learning, unsupervised learning and semi-supervised, and (3) the hybrid approach. (Madhoushi, Hamdan, & Zainudin, 2015) The lexicon dependent approach draws on two methods: (1) the dictionary-based method, which consists of detecting opinion seed words in the text and then finding synonyms/antonyms from a dictionary, and (2) the corpus-based method, which consists of finding opinion words in a corpus using a seed list of opinion words. (Madhoushi, et al., 2015, p. 289) Supervised learning approaches, which rely on existing classes in the training set for the classification of instances in the test set, typically draw on classifiers, such as Naïve Bayes, Support Vector Machine (SVM) and K-Nearest Neighbours (KNN). Unsupervised approaches draw on topic models (i.e.: clustering) such as LDA and pLSA. (Madhoushi, et al., 2015, p. 289) The hybrid approach uses a combination of methods from the two first approaches. In this project, we are interested in the lexicon-based approach and the corpus-based method: we use the Yelp Restaurant Sentiment Lexicon (Kiritchenko, et al., 2014b) to score sentiments in each review (the corpus).

Ganu, Elhadad and Marian (2009) used a sentiment-detection approach to derive ratings from text in user reviews and to make recommendations as an alternative approach to star ratings given by users. Their objectives were to accurately predict both the star ratings and the text ratings. (Ganu, et al., 2012) The analysts sorted **individual sentences** into restaurant-specific-domain classes (food, service, price, ambience, anecdotes, miscellaneous) and into sentiment classes (positive, negative, neutral, mixed). For the domain classification exercise, the analysts worked with three annotators for the manual annotation of a test sample and used SVM classifiers for the classification of the training sample. (Ganu, et al., 2012) For the sentiment classification exercise, they used a non-lexicon-based supervised approach. The comparison between star ratings and sentiment annotations produced by the classifier showed not only positive and neutral correlations, but also significant negative correlations (Pearson coefficient). Ganu et al. (2009) report that the results of their analysis show an improvement over the predictions produced by user ratings. Unfortunately, the journal article did not specify the programming language(s) used for classification.

In line with Ganu et al. (2012) above, Gan and Yu (2015) strive for an accurate detection of domain classes (food, service, ambience, pricing and special purpose) in sentence segments and for the effective determination of sentiment scores for each of these classes. They perform two exercises: topic classification and sentiment detection. For the topic classification exercise, the analysts hired restaurant experts to code 2000 user reviews and AAA reviews and to pick words/terms related to each domain topic. For the classification of review segments from the training dataset into the five domain topics, they used a supervised learning approach and a Naïve Bayes classifier. For the sentiment detection exercise, Gan and Yu (2015) used (1) a process for dealing with repeat letters in casual words and for dealing with misspellings, (2) a list of emoticons (:) , :-) , :(, :-(, etc.) and (3) the AFINN lexicon (F. A. Nielsen, 2011) to score sentiment from review sentences. Last but not least, Gan and Yu (2015) used the star rating given by the reviewer as the dependent variable and the weighted sentiment scores of the five domain topics as the independent variables. Among other things, they used ‘fast food’ as a control variable in their analysis and evaluation.

In line with the first phase of the two sentiment detection cases discussed above, Sajnani et al. (2013) sorted restaurant reviews into five classes (food, service, ambience, deals and worthiness). For the topic classification exercise, Sajnani et al. (2013) annotated reviews manually for the test set and extracted unigrams, bigrams and trigrams from the reviews in both the test and training sets. They selected to **remove special characters** from the review text, but chose **not to remove stop words** as some (i.e.: no, not) deliver negative sentiments. (Saini, 2013) We note that the analysts excluded fast food restaurants all together from the analysis, as fast food is a different domain than restaurants with table service.

In connection with a dataset (Yang, 2014) that combines Foursquared-tagged tweets crawled from Twitter and tips extracted from Foursquare, Yang et al. (2013) argue that the similarity between check-ins and the similarity between tips can improve the performance of restaurant recommendations. For the analysts, while the number of repeated check-ins by a user can translate into an implicit positive sentiment about a venue, a sentiment score can also be extracted from text-based tips using sentiment analysis techniques and express positive, neutral and negative sentiments. A 1-to-5-preference scheme was generated for both. The analysts used collaborative filtering as a base line and a location based social matrix factorization model to come up with improved recommendations. While the analysts used the NLTK toolkit (a Python package for natural language processing) and SentiWordNet3.0 (a sentiment lexicon) to process text in tips, the paper only provides algebra formulas, but no Python codes.

For lack of time, we choose NOT to pursue these approaches in this project except for the lexicon-based approach and the corpus-based method to score sentiments in restaurant reviews. Manual annotations of reviews require time and cross-validation between reviewers, which we don't have in this project. An unsupervised approach using clustering techniques could also have offered some kind of a breakdown of domain topics although the technique is known for its poor performance. In this project we focus more on the recommendation aspect of the problem, although it would have been nice to separate areas of satisfaction from dissatisfaction in the exercise. Another project perhaps? We also choose to use emoticons as relevant sentiment-expressing objects in reviews. They were available in both lexicons we selected (The AFINN lexicon we used in this project amalgamates two AFINN lexicons: one with emoticons only and one with commonly used words on the internet.).

d) Related works on text mining, natural language processing and sentiment detection packages

For this project, we consider the text-mining package **tm** for R (Feinerer, 2015; Feinerer, Hornick, & Artifex Software Inc., 2015) and the natural language processing package **NLTK** for Python (Bird, Klein, & Loper, 2015; NLTK Project, 2015) for the processing of text and the classification of review sentences. We also considered for the sentiment classification exercise the tm.plugin.sentiment package for R (Annau, 2015) and the SentiWordNet package for Python, (SentiWordNet, 2010) although we select for this project a sentiment detection approach using a text mining package and two lexicons - AFINN (Kuo, 2015; F. Nielsen, 2015; F. A. Nielsen, 2011) and the Yelp Restaurant Sentiment Lexicon (Kiritcheko, Zhu, & Mohammad, 2014; Kiritchenko, Zhu, Cherry, & Mohammad, 2014a; Kiritchenko, et al., 2014b) rather than a machine-learning based approach as the manual classification of review segments by domain topics will consume a major chunk of the time allocated for this project. Nielsen (2015) downloaded AFINN-en-165, a new version of the AFINN lexicon, on the GitHub in September 2015. An emoticon-based sentiment lexicon is also available on Neilsen's GitHub account. (F. Nielsen, 2015) We note that the Yelp Restaurant Sentiment Lexicon carries words with spelling errors where as the AFINN lexicon does not. Both offer conjugated verbs in present and past tense and singular and plural words. In this project, we use TweetTokenizer from NLTK to tokenize words and emoticons found in restaurant reviews.

3. The Description of the Data

We perform our classification and sentiment detection exercises mainly on the review dataset provided by the Yelp Dataset Challenge. (Yelp, 2015) The Yelp datasets cover 61,184 businesses in total, which are located in Las Vegas, Phoenix, Madison, Urbana, Charlotte, Waterloo (Canada), Pittsburg, Montreal (Canada), Edinburgh (UK) and Karisruhe (Germany). Below is a summary of the Yelp datasets. As the text in bold suggests, we are more specifically interested in Las Vegas restaurants although we can't ignore the Yelp restaurants (in italics) as they may provide key data for classification and recommendations in this project. We can't ignore the tourist factor. AFINN-en-165 offers both British and American spellings of verbs.

Reviews:	1,569,264 reviews / *990,627 restaurant reviews* / 434,491 restaurant reviews given unique reviewers of restaurants in Phoenix / **405,760 restaurant reviews in Las Vegas / 42,123 restaurant reviews in Las Vegas given unique reviewers of restaurants in Phoenix / 1,266 reviews of Thai restaurants in Las Vegas given unique reviewers of restaurants in Phoenix** /// 9,850 reviews of Thai restaurants in Phoenix given unique reviewers of restaurants in Phoenix / *11,428 reviews of Thai restaurants given unique reviewers of restaurants in Phoenix* / 10,911 reviews combined by unique users of Thai restaurants given unique reviewers of restaurants in Phoenix.
Businesses:	61,184 businesses / *21,892 restaurants* / 21,799 restaurants with reviews / **4,960 restaurants in Las Vegas / 3,837 restaurants in Las Vegas reviewed by unique reviewers of restaurants in Phoenix / 106 Thai restaurants in Las Vegas reviewed by unique reviewers of restaurants in Phoenix** /// 131 Thai restaurants in Phoenix reviewed by unique reviewers of restaurants in Phoenix / *319 Thai restaurants reviewed by unique reviewers of restaurants in Phoenix*.
Users:	366,715 users / *269,231 unique reviewers of restaurants* / 92,770 unique reviewers of restaurants in Phoenix / **129,019 unique reviewers of restaurants in Las Vegas.**
Tips:	495,107 tips / *304,388 restaurant tips* / **141,844 restaurant tips in Las Vegas.**
Check-ins:	45,166 check-ins / *18,640 restaurant check-ins* / **4,596 restaurant check-ins in Las Vegas.**

In this project, we concentrate on the Yelp Review and Business datasets: we choose to ignore the other Yelp datasets. Using filters, we convert the two datasets respectively into the Yelp Restaurant Review dataset and the Yelp Restaurant dataset, and again into the Las Vegas

Restaurant Review dataset and the Las Vegas Restaurant dataset. Restaurants in Las Vegas are included in the Yelp Restaurant Review dataset and the Yelp Restaurant dataset. The original Yelp datasets counted many other types of services and businesses with each its distinct domain topics. For this project, the working dataset was derived from the merger (left join) of the Yelp Restaurant dataset (right) to the Yelp Restaurant Review dataset (left) using business_id as the key.

In this project, we are mainly interested in the 'text' and the 'stars' features from the original Yelp Review dataset and the 'price_range' sub-feature (found in the 'attributes' column) from the original Yelp Business dataset. As provided by Yelp (Yelp, 2015), below is a list of features for both datasets :

REVIEW DATASET

```
'type': 'review',
'business_id': (encrypted business id),
'user_id': (encrypted user id),
'stars': (star rating, rounded to full-stars),
'text': (review text),
'date': (date, formatted like '2012-03-14'),
'votes': {(vote type)
```

Note: The 'funny', 'useful' and 'cool' columns were added into the working dataset from the unpacking of the votes column in the original dataset. The 'stars' column was renamed: 'review_ratings'. A new column was created to extract the weekday from the 'date' column: 'weekday'. The type column was dropped. Although Yelp claims the stars were rounded to half stars, the data was rounded to full stars.

BUSINESS DATASET

```
'type': 'business',
'business_id': (encrypted business id),
'name': (business name),
'neighborhoods': [(hood names)],
'full_address': (localized address),
'city': (city),
'state': (state),
'latitude': latitude,
'longitude': longitude,
'stars': (star rating, rounded to half-stars),
'review_count': review count,
'categories': [(localized category names)]
'open': True / False (corresponds to closed, not business hours),
'hours': {
    (day_of_week): {
        'open': (HH:MM),
        'close': (HH:MM)
    'attributes': {
        (attribute_name): (attribute_value), ...
```

Note: The 'stars' column was renamed: 'avg_stars'. Two new columns were created to extract the 'price_range' and the 'Waiter Service' values from the 'attributes' column. Several columns were added into the working dataset from the unpacking of the 'categories' column in the original dataset: binary values can tell us whether the restaurant falls under the specific category (ex.: Italian) or not. The 'type' column was dropped. There were 183 columns in the Yelp Restaurant dataset before the merge.

Below is a list of all the attributes in the working dataset: they can be used as filtering devices for narrowing the topic a notch further or as control variables for the validation of various classification results.

['review_id', 'user_id', 'business_id', 'name', 'date', 'weekday', 'review_ratings', 'text', 'votes', 'funny', 'useful', 'cool', 'hours', 'open', 'full_address', 'neighborhoods', 'city', 'state', 'latitude', 'longitude', 'review_count', 'avg_stars', 'attributes', 'price_range', 'WaiterService', 'categories', 'Bars', 'American (New)', 'Nightlife', 'Lounges', 'Restaurants', 'American (Traditional)', 'Burgers', 'Breakfast & Brunch', 'Cafes', 'Pubs', 'Irish', 'Chinese', 'Italian', 'Fast Food', 'Arts & Entertainment', 'Arcades', 'Food', 'Breweries', 'Food Delivery Services', 'Pizza', 'Gluten-Free', 'Asian Fusion', 'Soup', 'Sandwiches', 'Salad', 'Seafood', 'Sports Bars', 'Tapas/Small Plates', 'Coffee & Tea', 'Barbeque', 'Wine Bars', 'Mediterranean', 'Vegetarian', 'Health Markets', 'Specialty Food', 'Portuguese', 'Ice Cream & Frozen Yogurt', 'Diners', 'German', 'Delis', 'Bakeries', 'Beer, Wine & Spirits', 'Bistros', 'Chicken Wings', 'Hot Dogs', 'Greek', 'Sushi Bars', 'Japanese', 'Indian', 'Mexican', 'Tex-Mex', 'Bagels', 'Event Planning & Services', 'Caterers', 'Comfort Food', 'Tapas Bars', 'Cocktail Bars', 'Ethnic Food', 'Grocery', 'Steakhouses', 'Korean', 'Vietnamese', **'Thai'**, 'Middle Eastern', 'Tea Rooms', 'Hotels & Travel', 'Peruvian', 'Latin American', 'Creperies', 'French', 'Hotels', 'Taiwanese', 'Buffets', 'Southern', 'Cajun/Creole', 'Soul Food', 'Dive Bars', 'Dance Clubs', 'Caribbean', 'Fondue', 'Pakistani', 'Malaysian', 'Ethiopian', 'Jazz & Blues', 'Persian/Iranian', 'Hookah Bars', 'Active Life', 'Bowling', 'Donuts', 'Lebanese', 'Meat Shops', 'Venues & Event Spaces', 'Pool Halls', 'Spanish', 'Music Venues', 'Cheesesteaks', 'Fish & Chips', 'Salvadoran', 'Shopping Centers', 'Shopping', 'British', 'Kosher', 'Golf', 'Desserts', 'Convenience Stores', 'Vegan', 'Hawaiian', 'Drugstores', 'Cuban', 'Gastropubs', 'Russian', 'Juice Bars & Smoothies', 'Party & Event Planning', 'Fruits & Veggies', 'Karaoke', 'Seafood Markets', 'Performing Arts', 'Halal', 'Dim Sum', 'Cinema', 'Mongolian', 'Casinos', 'Social Clubs', 'Filipino', 'Argentine', 'Cantonese', 'Pretzels', 'Szechuan', 'Himalayan/Nepalese', 'Moroccan', 'African', 'Turkish', 'Afghan', 'Food Stands', 'Modern European', 'Food Trucks', 'Gas & Service Stations', 'Automotive', 'Irish Pub', 'Brazilian', 'Chocolatiers & Shops', 'Gelato', 'Hot Pot', 'Live/Raw Food', 'Health & Medical', 'Nutritionists', 'Brasseries', 'RV Parks', 'Singaporean', 'Airports', 'Colombian', 'Do-It-Yourself Food', 'Shaved Ice', 'Food Court', 'Swimming Pools', 'Pub Food', 'Ramen', 'Street Vendors', 'Local Services', 'Festivals', 'Ukrainian', 'Shanghainese', 'Venezuelan', 'Bubble Tea', 'Couriers & Delivery Services', 'Candy Stores', 'Izakaya']

Among these features, 'date', 'weekday', 'review_ratings', 'funny', 'useful', 'cool', 'review_count', 'avg_stars' and 'price_range' have numeric values and were used for the first descriptive statistics section below. Again, for the first recommendation exercise, this project focuses mainly on the ‘review_ratings’ feature, which is a restaurant rating (1 to 5) given by the user at the time the review was submitted. The first section of the descriptive section below aims to assess this data and to assess the correlation potential with other numeric attributes. For the sentiment detection exercise, this project focuses strictly on the Review ‘text’ feature and aims to extract from review segments sentiment scores that can in turn be used as ratings in a recommendation engine. We will look at the review text data in the second descriptive statistics section.

a) Descriptive Statistics – Review Ratings and other numeric features

The first round of assessments revealed a few null values among the numeric features: 'date', 'weekday', 'review_ratings', 'funny', 'useful', 'cool', 'review_count', 'avg_stars' and 'price_range'. Two instances originally from the Yelp Restaurant Review dataset were dropped as the user_id and values from other fields from the dataset were missing. This brought down the number of instances to 990,626. Out of these instances after the merge, 6,007 instances (0.6%) had no price_range: we filled in the missing data with the average price_range. We also converted in R the values in the ‘date’ column from ‘object’ to ‘date’ as saving the dataframe into a csv file in Pandas may have converted the data into an object. The weekday was computed in Pandas: we added one to the results as zero being Monday by default would not lend itself well to an assessment of the correlation between numeric attributes. Below in Tables 3.1, 3.2 and 3.3 are the summaries from Pandas and from R. Whereas Table 3.1 and Table 3.2 describe respectively Yelp Restaurant Reviews (after the merger) with a Pandas summary and Yelp Restaurant Reviews (after the merger) with an R summary, Table 3.3 describes Las Vegas Restaurant Reviews (after the merger) with an R summary. A correlation assessment between the numeric features follows in Table 3.4 and the distributions for each feature (other than votes and review count) follow in Figure 3.1 to Figure 3.5. The highest frequency in the votes and review count columns is by far ‘one’, followed by ‘two’ with a dramatic drop.

Table 3.1: Pandas Summary for Yelp Restaurant Reviews

	count	unique	top	freq	first	last	mean	std	min	25	50	75	max
review_id	990626	990626	nZUAzeJqhZyhoNALMy6fCA	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
user_id	990626	269231	ikm0UCahtK34LbLCEw4YTw	1086	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
business_id	990626	21799	4bEjOyTaDG24SY5TxsaUNQ	4137	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
date	990626	3409	2015-01-03	1319	2004-10-12	2015-01-08	NaN	NaN	NaN	NaN	NaN	NaN	NaN
weekday	990626	NaN	NaN	NaN	NaN	NaN	3.964108	2.043843	1	2	4	6	7
review_ ratings	990626	NaN	NaN	NaN	NaN	NaN	3.719636	1.264373	1	3	4	5	5
funny	990626	NaN	NaN	NaN	NaN	NaN	0.4432308	1.435056	0	0	0	0	141
useful	990626	NaN	NaN	NaN	NaN	NaN	1.0113	1.98802	0	0	0	1	166
cool	990626	NaN	NaN	NaN	NaN	NaN	0.5714871	1.575068	0	0	0	1	137
price_ range	990626	NaN	NaN	NaN	NaN	NaN	1.897303	0.7017934	1	1	2	2	4
review_ count	990626	NaN	NaN	NaN	NaN	NaN	348.0874	623.3128	3	56	143	339	4578
avg_ stars	990626	NaN	NaN	NaN	NaN	NaN	3.711793	0.5416396	1	3.5	4	4	5

OBSERVATIONS: We note that the review ratings have an average of 3.72 and a standard deviation of 1.26 whereas the average stars have an average of 3.71 and a standard deviation of 0.54. The average stars may be misleading, as they have not been computed by unique business_id whereas the review ratings were computed with a unique review_id. Therefore, the average and standard deviation computations for the restaurants in this dataset are dependent on the reviews and the duplication of some of the business_ids. The same applies to the price_range and the review_count as they were merged to the working dataset from the Yelp Restaurant dataset. We also note that the first review was performed on October 12, 2004 and the last on January 08, 2015. The highest frequency of restaurant reviews was reached on January 3, 2015 with 1,319 reviews in the day. Table 3.1 also identifies the most active reviewer with 1086 reviews in total and the most popular restaurant with 4137 reviews.

Table 3.2: R Summary for Yelp Restaurant Reviews

```
      X                            review_id                                user_id
 Min.   :     0   ---DgD3WlOJXFnTQP6H23A:     1  ikm0UCahtK34LbLCEw4YTw: 1086
 1st Qu.:247656   ---gFan7_-hicaSgAi86Hg:     1  kGgAARL2UmvCcTRfiscjug: 661
 Median :495313   ---l5p2MM0KIPWKhOy8SYg:     1  3gIfcQq5KxAegwCPXc83cQ: 657
 Mean   :495313   ---mAxs7gR3fRdQXjDn6cw:     1  DrWLhrK8WMZf7Jb-Oqc7ww: 638
 3rd Qu.:742971   ---mvnodbrJnF3GbBdkV5g:     1  glRXVWWD6x1EZKfjJawTOg: 586
 Max.   :990627   ---nwivGjbFDuTpspmA83A:     1  ia1nTRAQEaFWv0cwADeK7g: 560
                  (Other)               :990620  (Other)               :98643
8
                business_id         state               date
 4bEjOyTaDG24SY5TxsaUNQ:  4137   NV     :405759   Min.   :2004-10-12
 2e2e7WgqU1BnpxmQL5jbfw:  3517   AZ     :381614   1st Qu.:2011-07-10
 zt1TpTuJ6y9n551sw9TaEg:  3352   NC     : 65855   Median :2013-02-19
 Xhg93cMdemu5pAMkDoEdtQ:  2675   PA     : 46569   Mean   :2012-09-17
 sIyHTizqAiGu12XMLX3N3g:  2657   QC     : 34543   3rd Qu.:2014-03-26
 YNQgak-ZLtYJQxlDwN-qIg:  2619   WI     : 30852   Max.   :2015-01-08
 (Other)               :971669   (Other): 25434
    weekday      review_ratings      funny              useful
 Min.   :1.000   Min.   :1.00   Min.   :  0.0000   Min.   :  0.000
 1st Qu.:2.000   1st Qu.:3.00   1st Qu.:  0.0000   1st Qu.:  0.000
 Median :4.000   Median :4.00   Median :  0.0000   Median :  0.000
 Mean   :3.964   Mean   :3.72   Mean   :  0.4432   Mean   :  1.011
 3rd Qu.:6.000   3rd Qu.:5.00   3rd Qu.:  0.0000   3rd Qu.:  1.000
 Max.   :7.000   Max.   :5.00   Max.   :141.0000   Max.   :166.000

      cool            price_range     review_count      avg_stars
 Min.   :  0.0000   Min.   :1.000   Min.   :   3.0   Min.   :1.000
 1st Qu.:  0.0000   1st Qu.:1.000   1st Qu.:  56.0   1st Qu.:3.500
 Median :  0.0000   Median :2.000   Median : 143.0   Median :4.000
 Mean   :  0.5715   Mean   :1.897   Mean   : 348.1   Mean   :3.712
 3rd Qu.:  1.0000   3rd Qu.:2.000   3rd Qu.: 339.0   3rd Qu.:4.000
 Max.   :137.0000   Max.   :4.000   Max.   :4578.0   Max.   :5.000
```

OBSERVATIONS: We note that the review ratings have an average of 3.72 whereas the average stars have an average of 3.71. The average stars may be misleading, as they have not been computed by unique business_id whereas the review ratings were computed with a unique review_id. Therefore, the average and standard deviation computations for the restaurants in this dataset are dependent on the reviews and the duplication of some of the business_ids. The same applies to the price_range and the review_count as they were merged to the working dataset from the Yelp Restaurant dataset. We also note from this summary the most active reviewers and the most popular restaurants. Last but not least, the greater number of restaurant reviews was reached in Las Vegas (NV) with a count of 405,759 reviews and the second in Phoenix (AZ) with a count of 381,614 reviews.

Table 3.3: R Summary for Las Vegas Restaurant Reviews

```
       X                          review_id                                user_id    
 Min.   : 77487 ---gFan7_-hicaSgAi86Hg:     1  glRXVWWD6x1EZKfjJawTOg:    586  
 1st Qu.:265828 ---l5p2MMOKIPWKhOy8SYg:     1  ia1nTRAQEaFWvOcwADeK7g:    559  
 Median :463621 ---mAxs7gR3fRdQXjDn6cw:     1  Iu3Jo9ROp2IWC9FwtWOaUQ:    543  
 Mean   :506662 ---mvnodbrJnF3GbBdkV5g:     1  PV5voYSD43Cn_3gHmxG7DA:    472  
 3rd Qu.:735007 ---nwivGjbFDuTpspmA83A:     1  38JK-SfO9NkAGs1RwlH2Gw:    447  
 Max.   :990627 ---sV8KdwfBoDw38KW_WnQ:     1  9A2-wSoBUxlMd3LwmlGrrQ:    407  
                (Other)               :405753  (Other)               :402745  
                 business_id         state              date           
 4bEjOyTaDG24SY5TxsaUNQ:  4137   NV     :405759   Min.   :2004-10-26  
 2e2e7WgqU1BnpxmQL5jbfw:  3517   AZ     :     0   1st Qu.:2011-07-26  
 zt1TpTuJ6y9n551sw9TaEg:  3352   BW     :     0   Median :2013-02-26  
 Xhg93cMdemu5pAMkDoEdtQ:  2675   EDH    :     0   Mean   :2012-09-30  
 sIyHTizqAiGu12XMLX3N3g:  2657   ELN    :     0   3rd Qu.:2014-04-03  
 YNQgak-ZLtYJQxlDwN-qIg:  2619   FIF    :     0   Max.   :2015-01-08  
 (Other)               :386802   (Other):     0                       
    weekday      review_ratings      funny              useful        
 Min.   :1.000   Min.   :1.000   Min.   :  0.0000   Min.   :  0.0000  
 1st Qu.:2.000   1st Qu.:3.000   1st Qu.:  0.0000   1st Qu.:  0.0000  
 Median :4.000   Median :4.000   Median :  0.0000   Median :  0.0000  
 Mean   :3.907   Mean   :3.711   Mean   :  0.4685   Mean   :  0.9953  
 3rd Qu.:6.000   3rd Qu.:5.000   3rd Qu.:  0.0000   3rd Qu.:  1.0000  
 Max.   :7.000   Max.   :5.000   Max.   :141.0000   Max.   :166.0000  
```

```
      cool          price_range     review_count      avg_stars    
 Min.   :  0.0000   Min.   :1.000   Min.   :   3.0   Min.   :1.000  
 1st Qu.:  0.0000   1st Qu.:2.000   1st Qu.: 112.0   1st Qu.:3.500  
 Median :  0.0000   Median :2.000   Median : 287.0   Median :4.000  
 Mean   :  0.6019   Mean   :2.049   Mean   : 614.1   Mean   :3.704  
 3rd Qu.:  1.0000   3rd Qu.:2.000   3rd Qu.: 672.0   3rd Qu.:4.000  
 Max.   :137.0000   Max.   :4.000   Max.   :4578.0   Max.   :5.000  
```

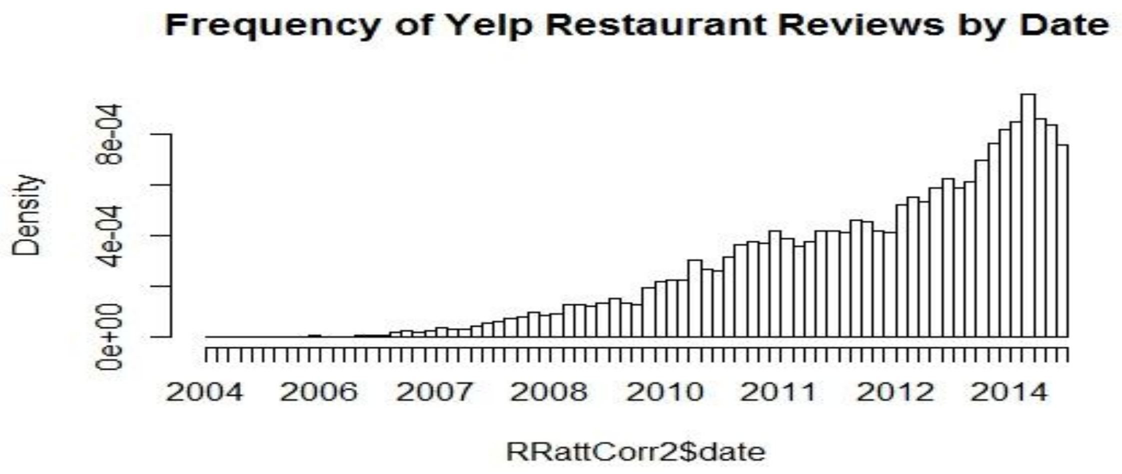
OBSERVATIONS: The first column, which is a row index from Pandas, informs us that the first Las Vegas restaurant review was found at line 77,487 in the Yelp Restaurant Review dataset. It has no specific meaning. We note that the review ratings have an average of 3.71 whereas the average stars have an average of 3.70. The average stars may be misleading, as they have not been computed by unique business_id whereas the review ratings were computed with a unique review_id. Therefore, the average and standard deviation computations for the restaurants in this dataset are dependent on the reviews and the duplication of some of the business_ids. The same applies to the price_range and the review_count as they were merged to the working dataset from the Yelp Restaurant dataset. Interestingly, the average price range for Las Vegas Restaurants is at 2.05 out of 5 whereas the average price range for Yelp Restaurants is 1.90. Perhaps, users in Las Vegas eat and drink a little more expensively although not by much: Is it the holiday factor? We also note from this summary the most active reviewers and the most popular restaurants. The top two restaurant reviewers in Las Vegas are found in the top six reviewers for all the Yelp Restaurant Reviews. The top six restaurants in Las Vegas are the same as the top six restaurants for the entire Yelp Restaurant Reviews. This may be an indication that some reviews in Las Vegas may have been entered by owners and/or by competitors. The first restaurant review entry took place on October 26, 2004 and the last on January 8, 2015.

Table 3.4: Correlation Assessment between Numeric Features in Yelp Restaurant Reviews

```
                  date        weekday  review_ratings    funny        useful
date           1.00000000  0.0482570869  0.01111988 -0.098828573 -0.120009471
weekday        0.04825709  1.0000000000 -0.01271297 -0.022952935 -0.023716182
review_ratings 0.01111988 -0.0127129708  1.00000000 -0.045741184 -0.038463162
funny         -0.09882857 -0.0229529347 -0.04574118  1.000000000  0.758073301
useful        -0.12000947 -0.0237161817 -0.03846316  0.758073301  1.000000000
cool          -0.11687440 -0.0245631607  0.05810034  0.815010079  0.852849877
price_range   -0.06659092 -0.0118682936  0.02116678 -0.004231098  0.010219792
review_count  -0.03916180 -0.0288715310  0.07089901  0.017342170 -0.006638168
avg_stars      0.08153575 -0.0009077439  0.39902995 -0.001445952  0.023100020
                   cool       price_range  review_count   avg_stars
date           -0.116874397 -0.066590918 -0.039161802  0.0815357491
weekday        -0.024563161 -0.011868294 -0.028871531 -0.0009077439
review_ratings  0.058100338  0.021166779  0.070899006  0.3990299455
funny           0.815010079 -0.004231098  0.017342170 -0.0014459523
useful          0.852849877  0.010219792 -0.006638168  0.0231000196
cool            1.000000000  0.004470324  0.017379378  0.0504211028
price_range     0.004470324  1.000000000  0.203090062  0.0428205904
review_count    0.017379378  0.203090062  1.000000000  0.1608884695
avg_stars       0.050421103  0.042820590  0.160888469  1.0000000000
```

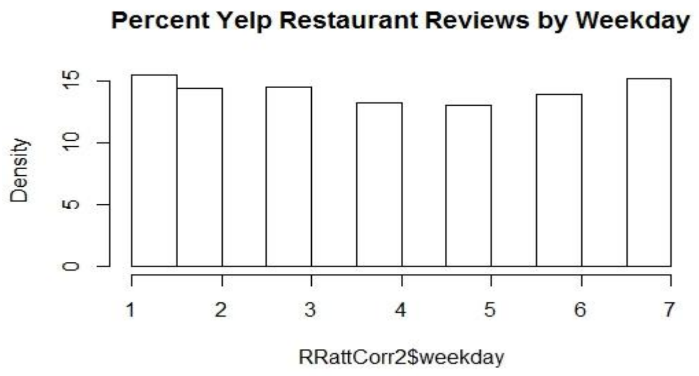
OBSERVATIONS: Other than votes (funny, useful and cool), the highest correlation is found between review_ratings and avg_stars at 0.39. This suggests that the average review rating for a restaurant provided by Yelp on their site has an effect on the reviewer's rating of the restaurant that is not very significant although the average review rating may have played a role in the selection of the restaurant being reviewed. The high correlation between votes (funny, useful and cool) is not particularly useful in this exercise other than they likely show similar distributions. Again, these votes may have played a role in the selection of the restaurant by users, but not so much in the rating of the restaurant. Furthermore, the correlation between the review_count and the avg_stars, which are both, extracted from the Yelp Restaurant dataset show little significance at 0.16.The same applies to the review_count and the price_range at 0.20. **There appears to be no significant correlation between review_ratings and the price_range at 0.02. Furthermore, there appears to be no significant correlation between the avg_stars and the price_range at 0.04. This is perhaps an argument against the assumption in this project that a user rating is a measure of satisfaction based on the quality/price ratio where price is automatically assumed in the rating. Nevertheless, we moving forward with the assumption as this ratio is a well-known concept in business.**

Figure 3.1: Distribution of Yelp Restaurant Reviews by date (124 bars between first and last entries)



OBSERVATIONS: This histogram shows that Yelp restaurant reviews have been increasing steadily from 2004 to 2015. The frequency of reviews for the peak period appears to be superior than 80,000. This peak period corresponds approximately to early September to early October 2014 as the last bar is likely from early December to early January (last entry January 8th, 2015). The histogram was built with 124 bars between October 14, 2004 and January 8, 2015.

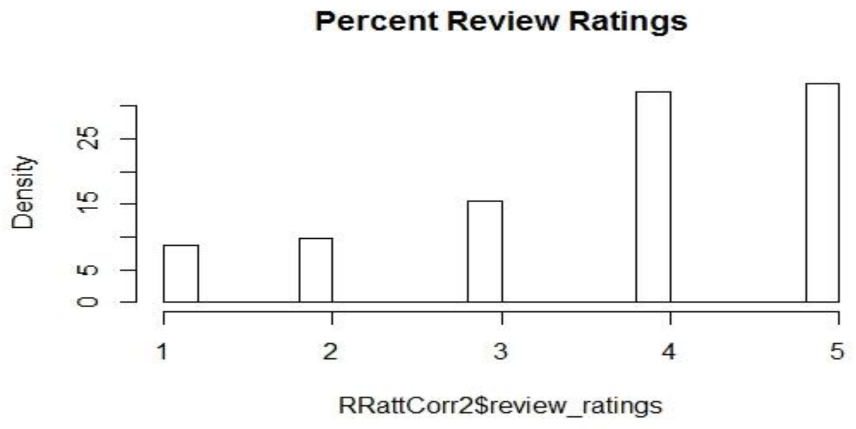
Figure 3.2: Percent of Yelp Restaurant Reviews per Weekday



```
   1       2       3       4       5       6       7
154,025 143,227 144,065 131,333 129,439 138,012 150,525
```

OBSERVATIONS: This histogram shows the percent restaurant reviews per weekday. 1 is Monday whereas 7 is Sunday. We added one (+1) to the Pandas datetime.weekday results for the correlation assessment above. Pandas assumes Sunday is 0. The highest percent is found on Sunday and Monday followed by Wednesday. This suggests that users rate and review restaurants more once Friday and Saturday are over.

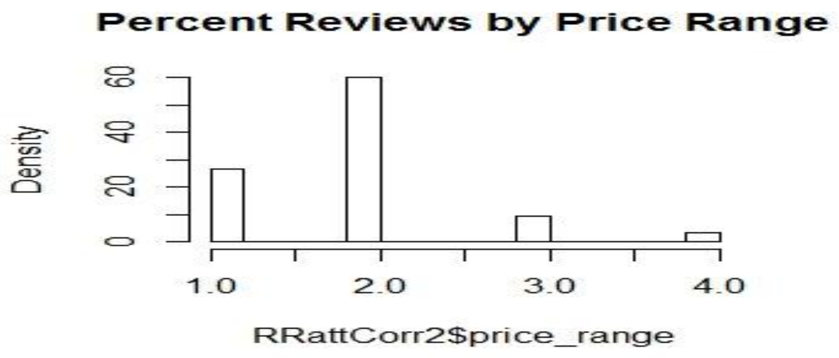
Figure 3.3: Distribution of Yelp Restaurant Review Ratings



```
     1         2       3        4        5
87,213  97,576 153,622 319,538 332,677
```

OBSERVATIONS: This histogram shows the percent review ratings from the Yelp Restaurant Review dataset. Most of the reviews have a rating of 4 or 5. This may be an indication that reviewers mostly review and rate their restaurant experiences when they are happy about their choice. They may also choose to rate their restaurant experience on the high side so that the restaurant will stay in business and so they will be able to repeat the experience. They may also tend not to write a review when the experience is OK (3) or mediocre (1) from fear of showing to friends they made a bad choice. Clearly, we do not have a normal distribution when it comes to ratings. A bias of some sort is apparent.

Figure 3.4: Distribution of Yelp Restaurant Reviews by Restaurant Price Range



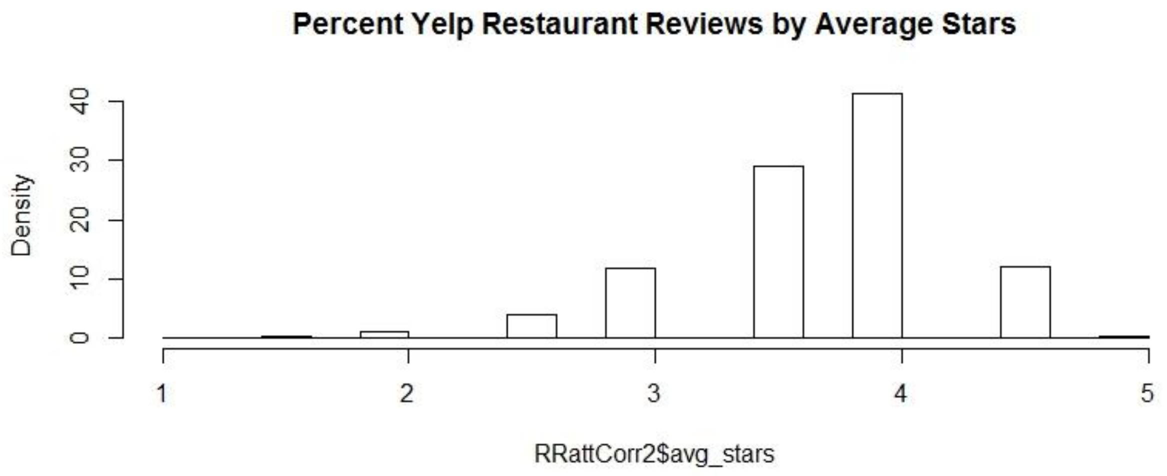
```
      1                  2                  3                 4
265,542            594,816             96,111            34,157
```

1 = under $10 2 = $11- $30 3 = $31- $60 4 = above $61

NOTE: price per person incl. 1 drink, tax and tip

OBSERVATIONS: This histogram shows the percent restaurant reviews by price range (from the Yelp Restaurant dataset). The great majority of reviews are for restaurants with a price range of 2, followed by restaurants with a price range of 1 and then 3. Fast food is included.

Figure 3.5: Distribution of Yelp Restaurant Reviews by Restaurant Average Stars



```
 1     1.5      2     2.5      3        3.5      4       4.5      5
372  2,620  10,220 39,593 118,047 286,735 409,803 119,662 3,574
```

OBSERVATIONS: This histogram shows the percent restaurant reviews by average stars (from the Yelp Restaurant dataset). The great majority of reviews are for restaurants with average stars of 4, followed by restaurants with average stars of 3.5. This may suggest that a majority of reviewers are more likely to choose to review restaurants within this average star range or more simply are more likely to choose restaurants within this star range in the first place. Nevertheless, the reviewers' restaurant ratings in the reviews (as seen in Figure 3.3) seem to be heavier in the 4 and 5 range. Might Yelp use a formula to adjust the average stars per restaurant that are advertised on its website?

b) Descriptive Statistics – Review Text

A pandas summary of the word counts is available in Table 3.5 and in Table 3.6 below. The word-count statistics in Table 3.5 were computed from individual reviews in the Yelp Restaurant Review dataset **from October 2004 to January 2015**. Using the same dataset, the word-count statistics in Table 3.6 were computed from **combining and grouping text reviews from October 2004 to January 2015 by restaurant** (business_id). The resulting dataframe shows that only 21,799 restaurants in the Yelp Restaurant Review dataset out of the 21,892 in the Yelp Restaurant dataset are taken into account. Some restaurants in the Yelp Restaurant Review dataset have no words; therefore, the discrepancy may be that 3 restaurants in total got no reviews and no ratings from reviewers. It is also possible that the 2 records we dropped in the data cleaning stage for lack of a proper user_id may account for some of the missing restaurants. From the R package 'stringi', we count the number of words in the text column. We acknowledge that the package underestimates by a small margin the number of words as some words are stuck together (ex.: I'm).

Table 3.5: Pandas Summary for Word Count in Review Text by Yelp Restaurant Review

	count	mean	std	min	25%	50%	75%	max
word_count	990,626	123	114	0	46	89	162	1,033

OBSERVATIONS: While the maximum number of words is 1,033 and the minimum is zero, the mean is 123 and the standard deviation is 114. After performing a query on the same dataframe, the business_id associated to the top word_count of 1,033 among all the Yelp restaurant reviews is: VVKXPSgU3KeLxrXVPYwStw. This business_id corresponds in the working dataset to *Yolie's Brazilian Steak House* in the Eastside of Las Vegas with a price range of 3, a review_count of 73 and avg_stars of 3. Its categories are Steakhouses and Brazilian. We note that these statistics are from the complete Yelp Restaurant Review dataset. This 1,000 word review was given by one us er.

Table 3.6: Pandas Summary for Word Count in Combined Review Text by Restaurant

	count	mean	std	min	25%	50%	75%	max
word_count	21,799	5,559	15,015	0	592	1,599	4,987	515,130

OBSERVATIONS: While the maximum number of words is 515,130 and the minimum is zero, the mean is 5,559 and the standard deviation is 15,015. After performing a query on the same dataframe, the business_id associated to the top word_count of 515,130 among all the Yelp restaurants is: 4bEjOyTaDG24SY5TxsaUNQ. This business_id corresponds in the working dataset to *Mon Ami Gabi* on the Strip in Las Vegas with a price range of 2, a review_count of 4,578 and avg_stars of 4. Its categories are Breakfast & Brunch, Steakhouses and French. Perhaps, it does not come as a surprise that this restaurant, which has the largest number of words in its reviews combined, is in Las Vegas.

An overview of the text column in the Yelp Restaurant Review dataset suggests that some records have reviews that were written in French (ex.: "J'adore faire mes courses avec un café…") and in German (ex.: "Wir wollten mal wieder griechisch essen..."). Reviews may have been written in other languages as well, perhaps even a mixture of languages. As we have selected to use the Yelp Restaurant Lexicon, whose units of analysis are restaurants in Phoenix (AZ), we have filtered out by default the great majority of reviews given in other languages although some sentences or some reviews may have remained. They will simply be ignored by the sentiment detection exercise using English lexicons.

Some typos (ex.: “it was **over priced** and not that great **shouldent** go again”) have been identified as well as letters in words that were duplicated several times for emphasis (ex.: “I'm **soooo** so thankful for authentic hotpot in town.”), but they have been surprisingly rare in the sample that was examined; therefore, a spell check does not seem to be necessary when using the AFINN lexicon. Words with duplicated letters will not be extracted when using the AFINN lexicon although they are likely to be when using the Yelp Restaurant lexicon (it has some). As the sample sentence suggests below, the use of smileys (and other symbols) are used in restaurant reviews; therefore, we choose lexicons with emoticons and the TweetTokenizer as stated earlier.

“ The food is really pretty inexpensive, so 10-20% for a table of two would usually only amount to about $2-4, so my advice is tip the waitresses well - they really deserve it! :) ”

CONCLUSIONS

The descriptive statistics show that no correlation exists between the numeric features of the working dataset. Although the statistics show no significant correlation between price and ratings and between price and average stars, we move forward with the assumption that price is assumed in the ratings given by users. The price distribution suggests that a large majority of reviewers select restaurant in the price range of 2 ($11 - $30 per person incl. one drink, tax and tip). Among Thai restaurants, this is also the case; however, there may be a higher percentage of restaurants with a price range of 1 than the Yelp dataset. We also have become aware that the average stars per business are lower than the reviewers’ ratings: this suggests that Yelp may be statistically adjusting the figures. The text descriptive statistics suggest that Las Vegas is a hot bed for reviews and marketing. It would also appear that a spell check may have already cleaned up the reviews: it is not easy to find obvious spelling errors.

4. The Approach

Two overviews of the same approach are provided below. Figure 4.1 shows two streams of actions that run simultaneously and merge together into a final comparison exercise. Table 4.1 lists the main steps for sentiment detection and Table 4.2 for recommender exercises.

Figure 4.1 : Diagram of the Approach for Tackling the Recommendation Problem

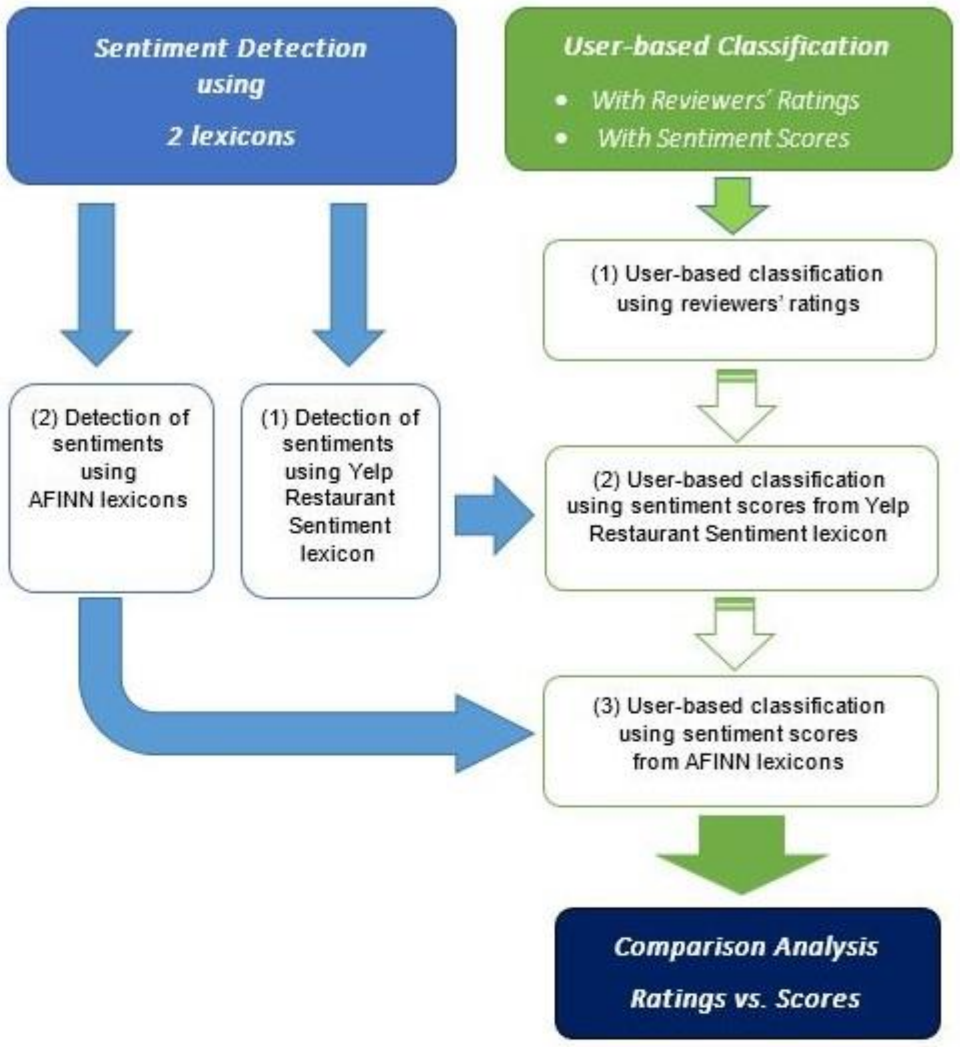

Sentiment Detection using 2 lexicons
User-based Classification
With Reviewers' Ratings
With Sentiment Scores
(1) User-based classification using reviewers' ratings
(2) Detection of sentiments using AFINN lexicons
(1) Detection of sentiments using Yelp Restaurant Sentiment lexicon
(2) User-based classification using sentiment scores from Yelp Restaurant Sentiment lexicon
(3) User-based classification using sentiment scores from AFINN lexicons
Comparison Analysis
Ratings vs. Scores

Table 4.1: Steps to Perform in Sentiment Detection

Detection of sentiments using lexicons
1. Use a natural language processing package (NLTK) and selected lexicons to extract sentiments for each review. 2. Evaluate the accuracy (NOT POSSIBLE IN THIS PROJECT AS THERE IS NO MEANS TO DO IT) Code sources: Kuo (2015), Neilsen (2015; 2011)

As far as sentiment detection is concerned, Neilsen (2015) provides on the Github not only the web-based lexicons he has built over the years, but also coding in Python to achieve the task. Kuo (2015) also offers a nice coding introduction to sentiment detection using Neilsen's AFINN lexicon. The analyst was unable to find a method to evaluate the accuracy of the sentiment score extracted from the two lexicons selected for this project.

Table 4.2: Breakdown of Steps in a Recommender Exercise

Prepare the data	• Define the problem: ○ **user-based** (user-to-user: users have similar tastes), or ○ item-based (item-to-item: items have similar ratings); ○ **using ratings** (1 to 5, where 1 = strongly dislike & 5 strongly like), or ○ using binary (0 or 1, where 0 = not known & 1 = known). • Cleanse and massage the data (input matrix) • Select a training set and a test set
Normalize	• Remove biases: Z-score, mean-centering, Log
Calculate the similarity (weights / neighbours)	• Calculate the Pearson correlation coefficient • Calculate the similarity with cosine similarity • Calculate the similarity with K-nearest neighbor (kNN)
Train	• Train the model (only in model-based approaches)
Predict	• Predict missing ratings • Produce the top-N predictions for every user
Denormalize	• Reverse the normalization (recommenderlab automatically executes this)
Evaluate the accuracy	• Provide accuracy (RMSE), precision, recall, F1, ROC and analysis.
Code sources: Hahsler (2015a), Rennie (2015), vakerkamachi (2015), Bigdata Doc (2014), Bhatnagar (2012), Caraciolo et al. (2013; 2011).	

Adapted from (Navisro Analytics, 2012)

Navisro Analytics' basic methodology presented in the adapted table above (Table 4.2) works in line with the features presented in the recommenderlab package for R. (Hahsler, 2015a) The problem definition section presents clearly what the 4 predictive options are in collaborative recommenders. For this project, we have selected, as Figure 4.1 suggests, the item-based approach using ratings/scores.

5. The Results

SUMMARY

One of the greatest disappointments in this project is the inability for *recommenderlab* to create and save ONE random set of training and test samples for adequate cross-evaluation of the accuracy of the models. The one sampling function (evaluationScheme) does it all, and every time you change a parameter the sample changes. To top it all off, the accuracy calculation function (calcPredictionAccuracy) generates an error if goodRating is NULL or non-existent in the sampling function (evaluationScheme). This goodRating parameter can only be assigned in the ONE and ONLY sampling function that is available. Another separate function should have been created to narrow down SUBSEQUENTLY the samples with other parameters, such as “given” and “goodRating”.

As a result of the algorithm’s vulnerability, it is nearly impossible to “adequately” assess the accuracy of the models. We did what we could. Like other analysts, we reverted to using simplistic non-random samples in some of the exercises in order to get at least some results, although the accuracy is far from being conclusive. Nevertheless, we got a sense that the Center approach and the Cosine method (Ratings, pt5 Scaled YRRL Scores) or the Center approach and the Jaccard method (Unscaled Scores, Scaled AFINN scores) were best suited for this project, considering the limited patterns of similarity extended to Las Vegas between users. On the other hand, the results from the confusion matrices (using ratings and sentiment scores alike) are similar across all models. Clearly, they show little evidence that the predicted ratings and scores extended to Las Vegas restaurants are promising unless at least one of the latter figures are among the top 5. Our sample users did not get any “meaningful” recommendations of restaurants in Las Vegas among the top 5. As far as confusion matrices are concerned, the consistent low accuracy from all the models suggests that none are better for the job.

In the end, based on RMSE results alone within the same notebook and model, our models were able to produce a list of top 5 Thai restaurants in Las Vegas for a few users. It is needless to say that the cold-start problem was indeed a challenge in this project as many users had given only one restaurant review. We chose not to eliminate these users so that we could understand the inner workings of *recommenderlab*. Although we discovered interesting facts along the way, those inner workings are still, for the most part, a mystery.

INTRODUCTION

In line with the approach we described in the diagram in Figure 4.1 above, we extracted the sentiments from reviews of Thai restaurants using two lexicons: the AFINN lexicon and the Yelp Restaurant lexicon. The AFINN lexicon we used combined English words and emoticons from two distinct AFINN lexicons whereas the Yelp Restaurant lexicon (YelpRRL) already had emoticons attached. Furthermore, the Yelp Restaurant lexicon included misspelled words where as the AFINN lexicon did not. In this project, we did not attempt to do a spell check on the reviews although rare spelling mistakes were found in a sample assessed by the analyst. The code for the detection of sentiments with these lexicons can be found in the Jupyter notebook (using Python) named "Soring Sentiments in Thai Restaurant Reviews with Lexicons.ipynb". Another Jupyter notebook by the name of "Making a smaller dataframe for R.ipynb" was also used prior to this phase to create a working dataset for the project.

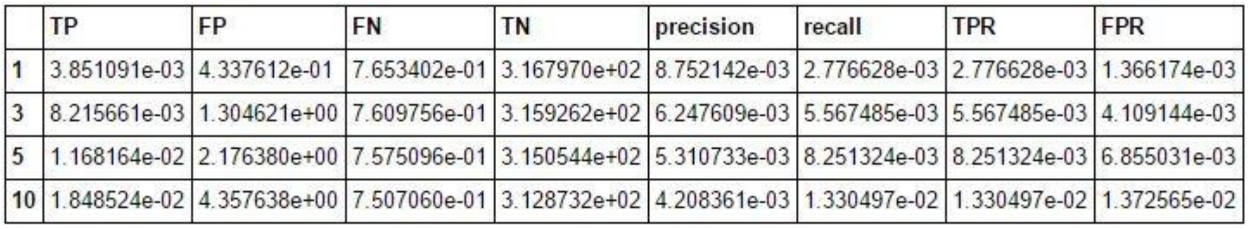
The third phase of the project brings us to the classification of ratings and sentiment scores using a user-based collaborative filtering approach (UBCF) and the *recommenderlab* package for R. The initial results using an item-based collaborative filtering approach (IBCF) were poor and non-promising, and the crab package for python is not up-to-date (last update 2011); as a result, we had to drop them. The code and the bulk of the analysis can be found on the following Jupyter notebooks (using R): "recommender with ratings.ipynb", "recommender with YelpRRL scores unscaled.ipynb", and "recommender with AFINN scores unscaled.ipynb". As we had anticipated more flexibility from *recommenderlab* with the creation of a random sample that could be used across notebooks and models, we also generated notebooks (NOT provided here as the text has yet to be edited) that performed the recommender tasks on YelpRRL scores scaled to 1 to 5 with 0.5 increments and on AFINN scores scaled to 1 to 5 with 0.5 increments following the Yelp rating's model. As it turns out, *recommenderlab* can perform its own scaling (ex: Z-score normalization).

The fourth phase follows with a comparison analysis between the top 5 restaurant recommendations based on ratings, the unscaled YelpRRL scores and the unscaled AFINN scores and accuracy results.

ANALYSIS

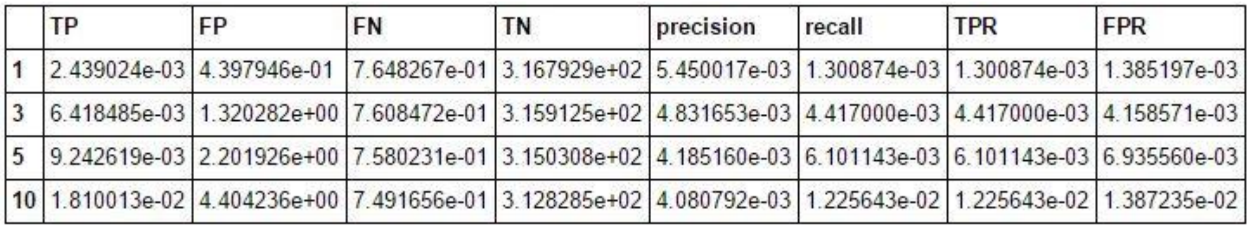
A comparison of the confusion matrices from the models coded with a consistent minimum rating / score (goodRating) as a baseline allows us to see the poor accuracy of the models. Again, the evaluationScheme function produced distinct training and testing samples from one model to another. We must take these accuracy results with a grain of salt.

Ratings (evaluationScheme = e6)



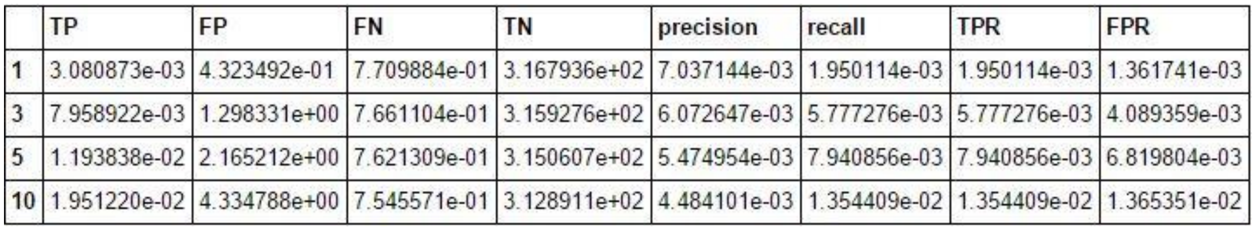
	TP	FP	FN	TN	precision	recall	TPR	FPR
1	3.851091e-03	4.337612e-01	7.653402e-01	3.167970e+02	8.752142e-03	2.776628e-03	2.776628e-03	1.366174e-03
3	8.215661e-03	1.304621e+00	7.609756e-01	3.159262e+02	6.247609e-03	5.567485e-03	5.567485e-03	4.109144e-03
5	1.168164e-02	2.176380e+00	7.575096e-01	3.150544e+02	5.310733e-03	8.251324e-03	8.251324e-03	6.855031e-03
10	1.848524e-02	4.357638e+00	7.507060e-01	3.128732e+02	4.208361e-03	1.330497e-02	1.330497e-02	1.372565e-02

YelpRRL Scores (evaluationScheme = e5)



	TP	FP	FN	TN	precision	recall	TPR	FPR
1	2.439024e-03	4.397946e-01	7.648267e-01	3.167929e+02	5.450017e-03	1.300874e-03	1.300874e-03	1.385197e-03
3	6.418485e-03	1.320282e+00	7.608472e-01	3.159125e+02	4.831653e-03	4.417000e-03	4.417000e-03	4.158571e-03
5	9.242619e-03	2.201926e+00	7.580231e-01	3.150308e+02	4.185160e-03	6.101143e-03	6.101143e-03	6.935560e-03
10	1.810013e-02	4.404236e+00	7.491656e-01	3.128285e+02	4.080792e-03	1.225643e-02	1.225643e-02	1.387235e-02

AFINN Scores (evaluationScheme = e7)

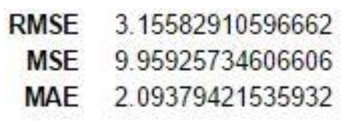


	TP	FP	FN	TN	precision	recall	TPR	FPR
1	3.080873e-03	4.323492e-01	7.709884e-01	3.167936e+02	7.037144e-03	1.950114e-03	1.950114e-03	1.361741e-03
3	7.958922e-03	1.298331e+00	7.661104e-01	3.159276e+02	6.072647e-03	5.777276e-03	5.777276e-03	4.089359e-03
5	1.193838e-02	2.165212e+00	7.621309e-01	3.150607e+02	5.474954e-03	7.940856e-03	7.940856e-03	6.819804e-03
10	1.951220e-02	4.334788e+00	7.545571e-01	3.128911e+02	4.484101e-03	1.354409e-02	1.354409e-02	1.365351e-02

The FP and the TN are fairly consistent from one model to the other. We note that the FP vary from 0.43 for N=1 to 4.33 for N=10 (approx.). We note that the TN are showing values of 317 for N=1 to 313 for N=10 (each evaluation Scheme has 319 columns (Thai restaurants)). The TP shows slightly better numbers among ratings than among scores although the difference is insignificant particularly in the N=10 range. Precision is highest with ratings although the difference is again insignificant particularly in the N=10 range. We are reminded that this is for all 319 Thai restaurants in the Yelp dataset. When a restaurant in Las Vegas is not found in the top 5, we are dealing with a decrease in accuracy and a more consistent value of inaccuracy among models. Therefore, from this perspective, taken into account that we have distinct samples from one model to another, no model is better than the other. The accuracy values found in other confusion matrices coded with a neutral value or with the median show no improvement again when N is high. The same applies to models using scaled sentiment scores (YelpRRL and AFINN). In short, from the perspective of confusion matrices, no model is better than the other.

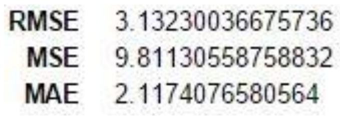
Again, RMSE, MSE and MAE calculations are hardly comparable from one recommender to another. We had selected early in the project to scale sentiment scores to 1-to-5 with 0.5 increments in case we needed at some point in the project to compare results between sentiment scores and user ratings. The latter were given by Yelp with a 1-to-5 scale and 0.5 increments, although ratings with 0.5 decimals attached are not easy to find. A comparison of the error results from the models with a consistent minimum rating / score as a baseline for the goodRating(s) allows us to see some difference in accuracy between the models. Again, the evaluationScheme function produced distinct training and testing samples from one model to another. We must take these accuracy results with a grain of salt.

Ratings (evaluationScheme = e6)



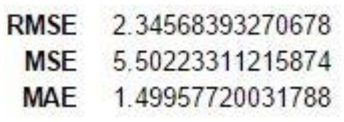
```
RMSE  3.15582910596662
 MSE  9.95925734606606
 MAE  2.09379421535932
```

Scaled YelpRRL Scores (evaluationScheme = e5(S))



```
RMSE  3.13230036675736
 MSE  9.81130558758832
 MAE  2.1174076580564
```

Scaled AFINN Scores (evaluationScheme = e7(S))

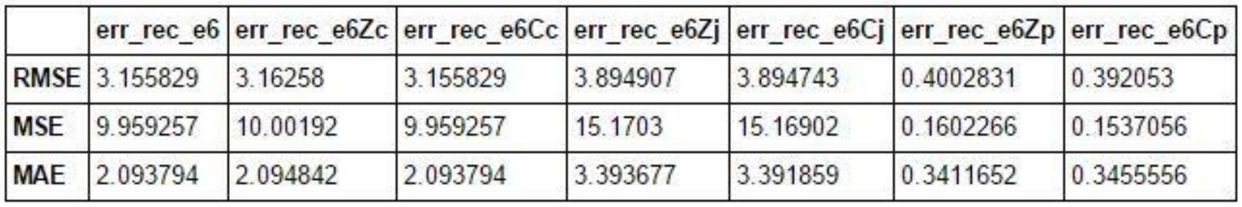


```
RMSE  2.34568393270678
 MSE  5.50223311215874
 MAE  1.49957720031788
```

These error results are misleading. For instance, as far as AFINN is concerned, the summary of the scaled scores produced a 1st quarter, a median and a 3rd quarter that were all equal to 2: no wonder the RMSE is so low! Too many reviews had a sentiment score of 2. The utility of the AFINN lexicon in this context can be questioned. The error values between the models using Ratings and the Scaled YelpRRL scores are similar although it appears that the error associated with ratings is higher. Again, these values can only be taken with a grain of salt as the randomly selected samples change from one model to the next. Again, it is impossible from this data to determine which model provides the greatest accuracy. From a timesaving perspective, this suggests that ratings alone are good enough. Sentiment detection steps are simply added on top of the recommender code. If they add no value, they have no utility.

This inconclusive comparison between models across ratings and sentiment scores above brings us to an analysis of the accuracy of various normalization approaches (Center (C), Z-score (Z)) and similarity assessment methods (cosine (c), Jaccard (j), Pearson (p)) in the delivery of predictions with the same sample and within the same rating/score prediction model. The purpose of this paragraph is to compare errors to one another in the same table, and not so much from one table to another (again a question of sample). Cc stands for Center and cosine, Zj for Z-score and Jaccard, and so forth. These abbreviations are attached at the end of the column name.

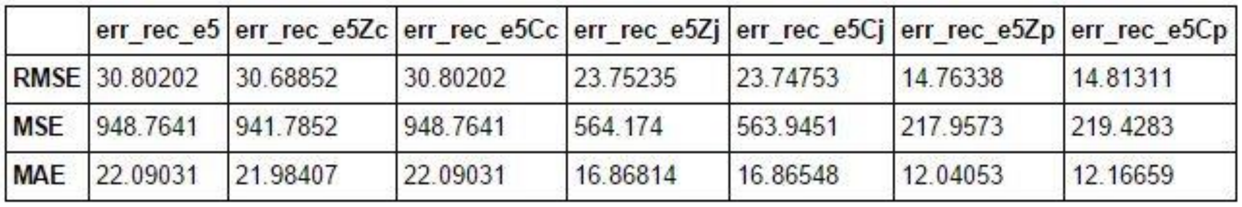
Ratings (evaluationScheme = e6)



	err_rec_e6	err_rec_e6Zc	err_rec_e6Cc	err_rec_e6Zj	err_rec_e6Cj	err_rec_e6Zp	err_rec_e6Cp
RMSE	3.155829	3.16258	3.155829	3.894907	3.894743	0.4002831	0.392053
MSE	9.959257	10.00192	9.959257	15.1703	15.16902	0.1602266	0.1537056
MAE	2.093794	2.094842	2.093794	3.393677	3.391859	0.3411652	0.3455556

The best accuracy results for ratings (e6 sampling) were produced by the Recommender function using parameters Center and Pearson; however, this method did not produce any predictions for Las Vegas restaurants. We therefore chose for our sample user the parameters Center and cosine, which are the default values for “Recommender”, the function.

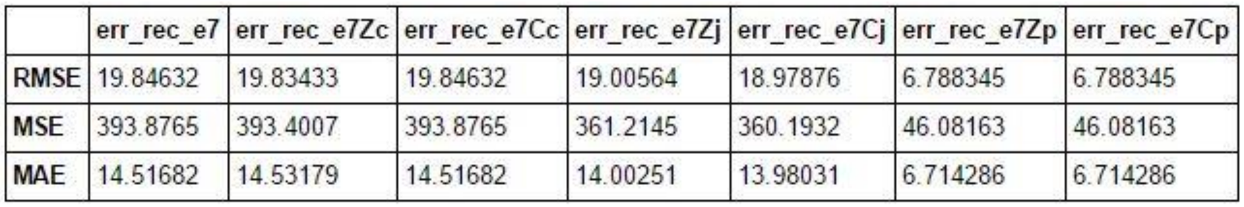
YelpRRL Scores (evaluationScheme = e5)



	err_rec_e5	err_rec_e5Zc	err_rec_e5Cc	err_rec_e5Zj	err_rec_e5Cj	err_rec_e5Zp	err_rec_e5Cp
RMSE	30.80202	30.68852	30.80202	23.75235	23.74753	14.76338	14.81311
MSE	948.7641	941.7852	948.7641	564.174	563.9451	217.9573	219.4283
MAE	22.09031	21.98407	22.09031	16.86814	16.86548	12.04053	12.16659

The best accuracy results for YelpRRL Scores (e5 sampling) were produced by the Recommender function using parameters Z-score and Pearson; however, this method did not produce any predictions for Las Vegas restaurants. Consequently, we chose for our sample user the parameters Center and Jaccard. Compared to Pearson, Jaccard delivers results fast.

AFINN Scores (evaluationScheme = e7)



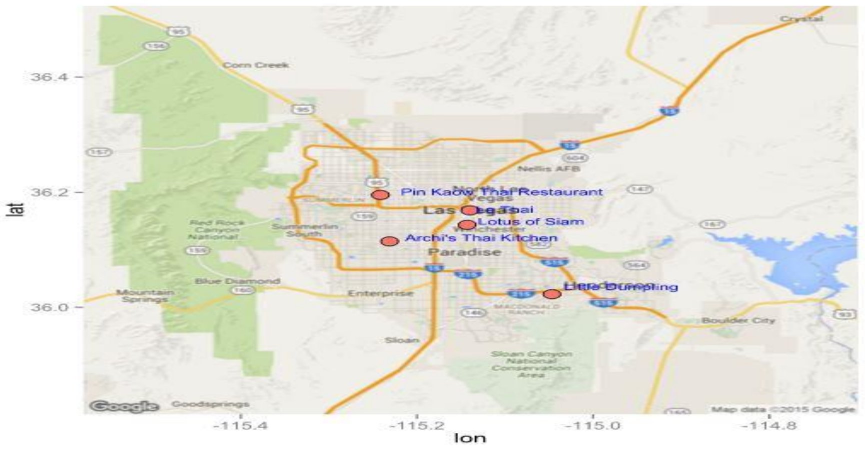
	err_rec_e7	err_rec_e7Zc	err_rec_e7Cc	err_rec_e7Zj	err_rec_e7Cj	err_rec_e7Zp	err_rec_e7Cp
RMSE	19.84632	19.83433	19.84632	19.00564	18.97876	6.788345	6.788345
MSE	393.8765	393.4007	393.8765	361.2145	360.1932	46.08163	46.08163
MAE	14.51682	14.53179	14.51682	14.00251	13.98031	6.714286	6.714286

The best accuracy results for AFINN Scores (e7 sampling) were produced by the Recommender function using parameters Z-score / Center and Pearson; however, these methods did not produce any predictions for Las Vegas restaurants. As a result, we chose for our sample user the parameters Center and Jaccard. Whereas Pearson is computationally greedy, Jaccard performs the best from a resource perspective.

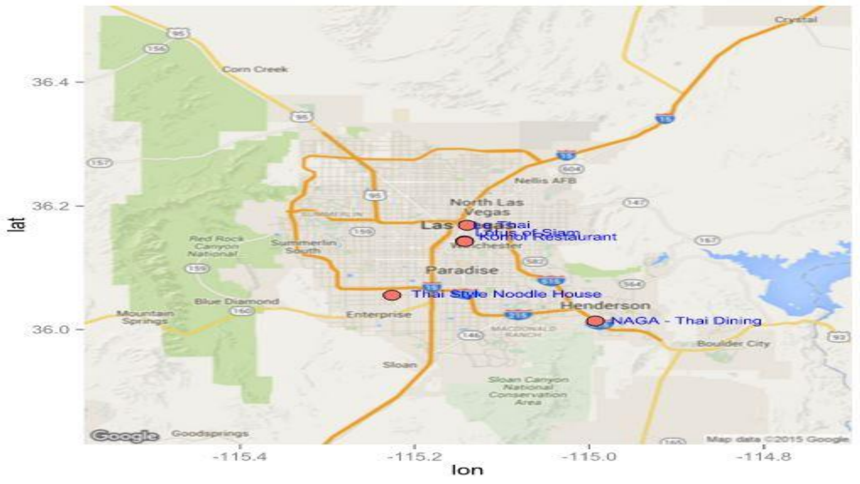
RECOMMENDING RESTAURANTS

For recommendations to individual users (User 1 of distinct user samples), we used a set of users who had a minimum of one predicted rating / score associated to a restaurant in Las Vegas. Issues arose with single ratings: *recommenderlab* is a little finicky and provides as recommendations to this type of user the first five restaurants on the list with an average rating / score prediction across restaurants. Perhaps, these single-review users should have been filtered out early in the notebook. We selected not to in order to get acquainted with some of the key features of *recommenderlab* and with some of its assumptions. Below are the five restaurants recommended to User 1 of each sample.

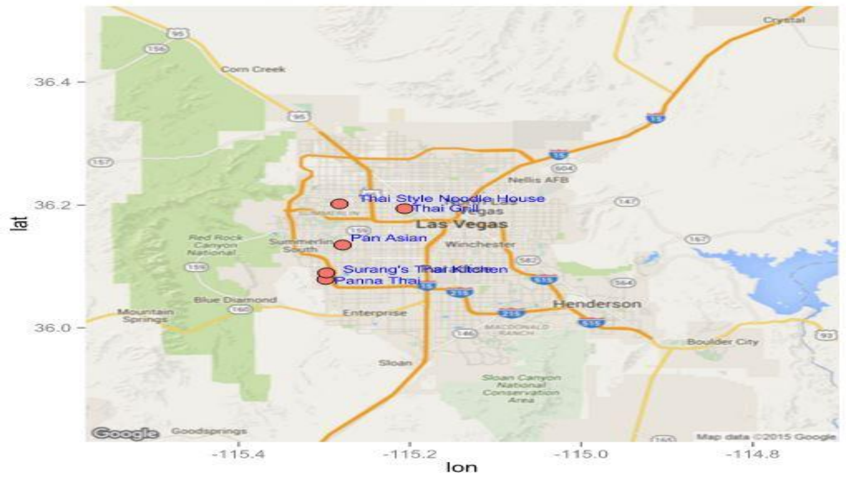
from Ratings (evaluationScheme = e6)



from YelpRRL Scores (evaluationScheme = e5)



from AFINN Scores (evaluationScheme = e7)



Again, we chose users that are different from one another to emphasize the fact that we could NOT use the same random samples for predictions across models. This is a major vulnerability of the *recommenderlab* package. Otherwise, it performs well and reliably. All Thai restaurants recommended fell in the price range of 1 or 2. Among some of the restaurants reviewed, we came across one restaurant that had been rated poorly, reviewed negatively according to YelpRRL and reviewed positively according to AFINN by the same user. It had no waiter service and was located at Las Vegas International airport. To the analyst's surprise, *recommenderlab* did not produce any meaningful recommendations of restaurants to one of the sample users who reviewed this restaurant in spite of two restaurant reviews. The analyst still does not understand fully many of the assumptions and reactions of *recommenderlab*. Its inner workings are a mystery.

CONCLUSION

Our hopes and expectations for this project were excessive. Crab, the recommender engine for Python, is out of date and cannot be reliably used. The *recommenderlab* package for R works reliably well and produces consistent outputs. Surprisingly, the package performs well with large numbers and decimals included. The analyst initially had a concern about patterns of similarity between users; however, this was not a problem. Scaling was in fact unnecessary for the recommendation engine to deliver. However, *recommenderlab* was clearly not designed for cross-evaluations of models with the same random sample. As a result, with the limitations we established early in this project, we cannot tell from our findings whether sentiment scores can improve top recommendations given to a user. We can only give them, as we can give recommendations from user ratings. Sentiment scores and ratings produce different recommendations, as the data is quite different from the start. The bell-shape curve of the sentiment scores extracted with the YelpRRL did NOT improve the accuracy. For the most part, the accuracy seem to reach its highest potential with the top 3 restaurants predicted by some models. As the working dataset had been narrowed down to users who had given at least one review to a restaurant located in Phoenix, the many restaurants in Las Vegas that had been rated by others were not included. Had they been included, the accuracy of predictions and the quality of recommendations "from ratings" may have improved. The analyst felt that this could not be done with sentiment scores as the development of the YelpRRL was based on Phoenix restaurants exclusively. Only reaching beyond the stars…

DISCUSSION

Using Jupyter notebook was very useful in many respects. Everything is there, all the code and the text, except perhaps the environment data (.RData). It is possible that, if we had this visually available, that the analyst could have found a way to keep the same random sample vertically as well as horizontally without having to convert the data to save it. However, after careful reflection, it still does not appear to be possible. A reengineering of the sampling function into two or three separate functions may resolve the dilemma. It might improve the research capability of the package. On the other hand, with more experience with the package and statistical tools, the analyst may uncover the key to the mystery.

Sources

Annau, Mario. (2015). tm.plugin.sentiment: github. Retrieved from https://github.com/mannau/tm.plugin.sentiment

Bhatnagar, Saurabh (saurabhat). (2012, September 26). Testing recommender systems in R. Retrieved from http://www.r-bloggers.com/testing-recommender-systems-in-r/

Bigdata Doc. (2014, September 25). Recommender Systems 101 - a step by step practical example in R. Retrieved from bigdata-doctor.com/recommender-systems-101-practical-example-in-r/

Bird, Steven, Klein, Ewan, & Loper, Edward. (2015). Natural Language Process with Python - Analyzing Text with the Natural Language Toolkit Retrieved from http://www.nltk.org/book/

Caraciolo, Marcel. (2013, October 22). Introduction to Non-Personalized Recommenders. *recsys* Retrieved September 23, 2015, from https://github.com/python-recsys/recsys-tutorial/blob/master/tutorial/0-Introduction-to-Non-Personalized-Recommenders.ipynb

Caraciolo, Marcel, Melo, Bruno, & Caspiro, Ricardo. (2011, July 11 - 16, 2011). *Crab: A Recommendation Engine Framework for Python.* Paper presented at the The 10th Python in Science Conference (SciPy 2011), Austin, Texas.

Carrillo, Naomi, Elmaleh, Idan, Gallego, Rheanna, Kloock, Zack, Ng, Irene, Perez, Jocelyne, . . . Shiroma, Ryan. (2013). Recommender Systems Designed for Yelp.com. 8.

Feinerer, Ingo. (2015). Introduction to the tm Package: Text Mining in R. 8. Retrieved from https://cran.r-project.org/web/packages/tm/vignettes/tm.pdf

Feinerer, Ingo, Hornick, Kurt, & Artifex Software Inc. (2015). Package 'tm': Text Mining Package. *CRAN*, 58. Retrieved from https://cran.r-project.org/web/packages/tm/tm.pdf

Gan, Qiwei, & Yu, Yang. (2015, January 5-8, 2015). *Restaurant Rating: Industrial Standard and Word-of-Mouth -- A Text Mining and Multi-dimensional Sentiment Analysis.* Paper presented at the 2015 48th Hawaii International Conference on System Sciences (HICSS), Hawaii, USA.

Ganu, Gayatree, Elhadad, Noémie, & Marian, Amélie. (2009). *Beyond the Stars: Improving Rating Predictions using Review Text Content*. Paper presented at the Twelfth International Workshop on the Web and Databases (WebDB 2009), Providence, Rhode Island. http://spidr-ursa.rutgers.edu/resources/WebDB.pdf

Ganu, Gayatree, Kakodkar, Yogesh, & Marian, Amélie. (2012). Improving the quality of predictions using textual information in online user reviews. *Information Systems, 38*(1), 15. doi: 10.1016/j.is.2012.03.001

Hahsler, Michael. (2015a). Package 'recommenderlab' Retrieved from https://cran.r-project.org/web/packages/recommenderlab/recommenderlab.pdf

Hahsler, Michael. (2015b). recommenderlab: A Framework for Developing and Testing Recommendation Algorithms (vignette)   Retrieved from https://cran.r-project.org/web/packages/recommenderlab/vignettes/recommenderlab.pdf

Jannach, Dietmar, Zanker, Markus, Felfernig, Alexander, & Friedrich, Gerhard. (2011). *Recommender systems an introduction*. Cambridge: Cambridge University Press.

Kiritcheko, Svetlana, Zhu, Xiaodan, & Mohammad, Saif M. (2014). Sentiment Analysis of Short Informal Texts. *Journal of Artificial Intelligence Research, 50*(2014), 39.

Kiritchenko, Svetlana, Zhu, Xiaodan, Cherry, Colin, & Mohammad, Saif M. (2014a, August 2014). *NRC-Canada-2014: Detecting Aspects and Sentiment in Customer Reviews.* Paper presented at the Eigth International Workshop on Semantic Evaluation Exercises (SemEval-2014), Dublin, Ireland.

Kiritchenko, Svetlana, Zhu, Xiaodan, Cherry, Colin, & Mohammad, Saif M. (2014b). Yelp Restaurant Sentiment Lexicon (National Research Council of Canada, Trans.): Mohammad, Saif M.

Kuo, Wei-Ting. (2015). *Sentiment Analysis by NLTK*. Paper presented at the Pycon Apac 2015 Back to_future_II, Taipei, Taiwan. http://fr.slideshare.net/waitingkuo0527/sentiment-analysisbynltk

Madhoushi, Zohreh, Hamdan, Abdul Razak, & Zainudin, Shaila. (2015, July 28-30). *Sentiment analysis techniques in recent works.* Paper presented at the Science And Information Conference 2015 (SAI), London, UK.

Navisro Analytics. (2012). Recommender Systems: fr.slideshare.net.

Nielsen, Finn. (2015, October 4). AFINN sentiment analysis with Python. Retrieved from https://github.com/fnielsen/afinn

Nielsen, Finn Arup. (2011). AFINN (AFINN-111, AFINN-96 ed.). Lyngby, Denmark: dtu.dk.

NLTK Project. (2015, September 5, 2015). NLTK 3.0 documentation Retrieved October 3, 2015, from http://www.nltk.org/

Rennie, Matthew. (2015, October 22). yelp-ml-reco-engine. Retrieved from https://github.com/matthewrennie/yelp-ml-reco-engine/blob/master/recommender.R

Saini, Vaibhav. (2013). Classifying Yelp reviews into relevant categories Retrieved October 2, 2015, from http://www.ics.uci.edu/~vpsaini/

Sajnani, Hitesh, Saini, Vaibhav, Kumar, Kusum, Gabrielova, Eugenia, Choudary, Parmit, & Lopes, Cristina. (2013). Classifying Yelp reviews into relevant categories (pp. 15): University of California, Irvine.

SentiWordNet. (2010). SentiWordNet Retrieved October 3, 2015, from http://sentiwordnet.isti.cnr.it/
varkerkamachi. (2015, October 22). yelp_recommender. Retrieved from https://github.com/varkerkamachi/yelp_recommender

Yang, Dingqi. (2014). Foursquare Dataset. *Dingqi Yang's Homepage* Retrieved September 18, 2015, from https://sites.google.com/site/yangdingqi/home/foursquare-dataset

Yang, Dingqi, Zhang, Daqing, Yu, Zhiyong, & Zhu, Wang. (2013, May 1-3, 2013). *A Sentiment-enhanced Personalized Location Recommendation System.* Paper presented at the 24th ACM Conference on Hypertext and Social Media (HT '13), Paris, France.

Yelp. (2015). Yelp Dataset Challenge Retrieved 17 September, 2015, from http://www.yelp.com/dataset_challenge