STA242 Prof. Duncan Temple Lang

Design Personal Ratings and Recommendations for Yelp

Liu, Yunying (Yuki)
yukliu@ucdavis.edu

Shi, zhewen
zwshi@ucdavis.edu

Cao, Jiexuan
jxucao@ucdavis.edu

June 10, 2015

SSH: git@bitbucket.org:Yuki0425/sta242_project.git

1 Abstract

In order for business owners to predict how users will respond to their products or service, we must uncover the tastes of the users and the properties of the product/service. For example, in order to predict whether there will be an increasing amount of customers give positive reviews to a local *Thai restaurant*, it helps to understand that thai food is known for its complex interplay of *sour*, *sweet* and *spicy* and the dishes often contain *coconut flavor* and *curry*, as well as the customers' level of interest in oriental cuisines. *User feedback* is required to discover these dimensions, which comes in the forms of *ratings* and *reviews.* Moreover, whether that *Thai restaurant* is located *downtown* or close to a *university*, whether it has *parking* or *wifi* and other business attributes may also play a role into forecasts about its business. In this project, we describe methods for inferring business' star level, users' taste, and the positive and negative features about the business ususing statistical models and sentiment analysis of Yelp reviews. All metadata aboout Yelp reviews, business information, and Yelp user information are acquired through Yelp Data Challenge [1] as well as using Python API streaming.

2 Introduction and Data Description

Yelp kicked off in 2004 and had been publishing crowd-sourced reviews about local business. The company also trains small business to respond to reviews responsibly, host social events for reviewers, and provides data about business, such as health scores. As of 2014, Yelp.com has 135 million monthly visitors and 71 million reviews. 85 percent of small business listed on the site have a rating of three stars or better, but some negative reviews are very personal or extreme. Every user can give a review a "thumbs-up" if it is "userful", "funny", or "cool".

The very rich Yelp-provided dataset used in this project has five `json` files that include *business information, check-in hours, reviews, tips details,* and *user information. Business records* has 15 subcategories:

```
'type', 'business_id', 'name', 'neighborhoods', 'full_address', 'city',
'state', latitude', 'longtitude', 'stars', 'review_count',
'categories', 'open', 'hours', 'hours'.
```

Reviews has 11 categories:

```
'type', 'user_id', 'name', 'review_count', 'average_stars', 'votes',
'friends', 'elite', 'yelp_since', 'compliments', 'fans'.
```

[1] The dataset can be downloaded from: *http://www.yelp.com/dataset_ challenge*

Check-in records include `'type'`, `'business_id'`, `'checkin_info'`. *Tip* information has `'tip'`, `'text'`, `'business_id'`, `'user_id'`, `'date'`, `'likes'`.

Figure 2.1 shows distributions of restaurants in Phoenix, Arizona and reviews of restaurants there. It is clearly that most restaurants are in business areas with heavy traffic. Reviews are more concentrated in business district.

The entire date contains over 1.5M reviews and 500K tips by 366K users for 61K business. Moreover, the datasets offer quite a wide range of business types from small local family owned restaurants to world's well known department stores, from health & medical to auto repair, from SPA and salons to church and religion organizations. This report will mainly focus on the 990k reviews from 21k different restaurants in top 10 most popular cities. The majority of our work involves generating sets of features to use in our model. We will describe two methods: statistical data mining and sentiment analysis on user-provided reviews. Moreover, we will run feature selection methods to pick out the best features in the later model selection section and experimental results section.

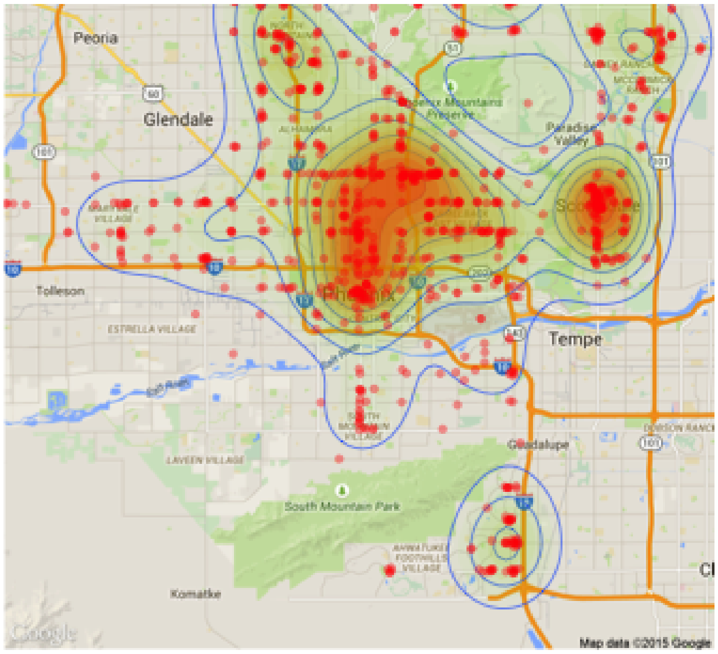


Figure 2.1: Heatmap showing number of reviews of restaurants in Pheonix, AZ. Red spots depict areas with large number of overal reviews

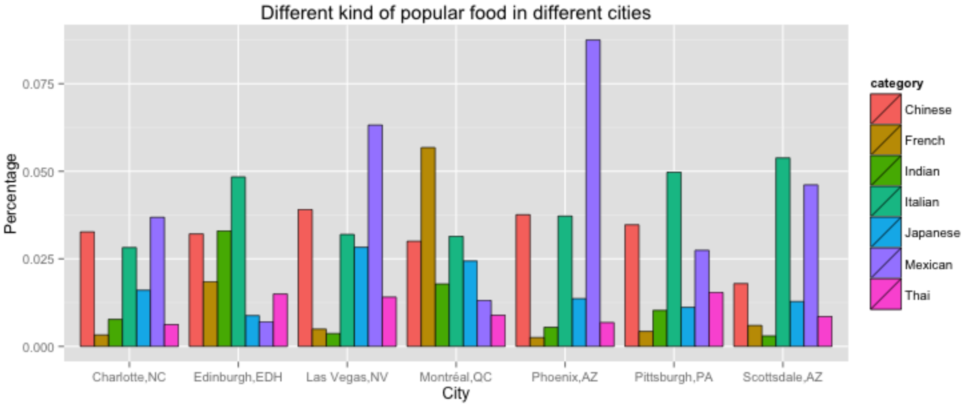
3 Basic Data Analysis and Data Manipulation

In this section, we identify two categories of features. The first category contains features which describe metadata about business locations, including longtitude and latitude, whether the restaurant is family friendly or has wifi, the number of business within 2 miles radius that share a category. The second categories mainly carry features about users' taste. In the data set, restaurants are distributed in 270 cities. We first sort the cities by its number of restaurants on Yelp and select the first seven popular cities with at least 200 restaurnts to be our study subjects.

3.1 Business in Different Cities

Each city might have its own style. Our goal is to find difference between restaurants in different cities and the hidden pattern behind data. We extract categories of all restaurants in these seven cities("Charlotte,NC", "Edinburgh,EDH", "Las Vegas,NV", "Montreal,QC", "Phoenix,AZ", "Pittsburgh,PA" and"Scottsdale,AZ") and look at the number of restaurants under different subcategories.

Figure 3.1: Different kind of popular food



From 3.1, we can see that percentage of Chinese restaurants are stable in these cities. However, Phoenix has many Mexican restaurants. Edinburgh and Montreal have few Mexican restaurants. Since most people like food in their home countries, we can conclude that chinese people travel and live in many cities in the world. Mexican people like America most. In America, the closer to Mexico, the larger number of Mexican the city has.

If we only take Thai food for example, we can see from Figure 3.2 and Figure 3.3 that stars of restaurants have approximate normal distribution in each city. But reviews do not have that distribution. Star of many reviews are greater than 3. There are two reasons. One is that most reviews like restaurants with high star levels. The other is that most people are kind hearted and do not want to post negative reviews on the Internet. From Figure 3.4, we know that it is the first reason, because most reviews are about restaurants which have 3 or 4 star level . There is another interesting thing that most users of Yelp are american. In Edinburgh and Montreal, there are less reviews.

3.2 Users in Different Cities

Let's take uses in Peonix for an example. There are 45867 distinct users in Pheonix and each of them has a *user's star* that implies their creditbility on Yelp. Below is some basic information about users in Pheonix

Figure 3.2: Stars of Thai food in different cities

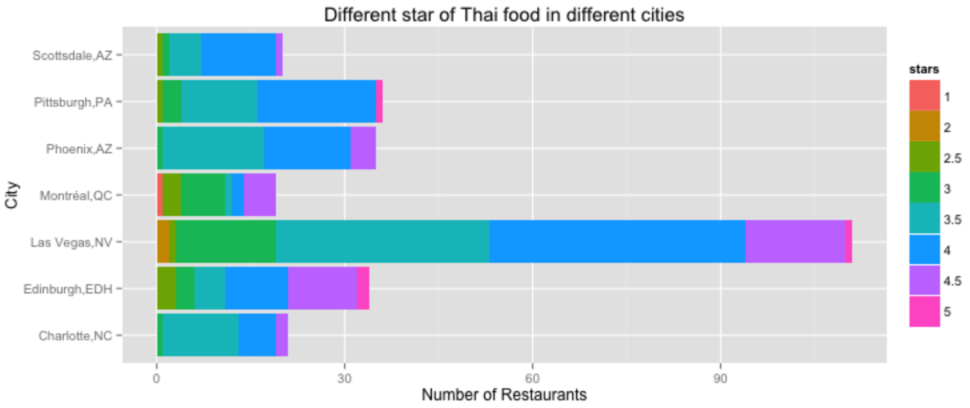
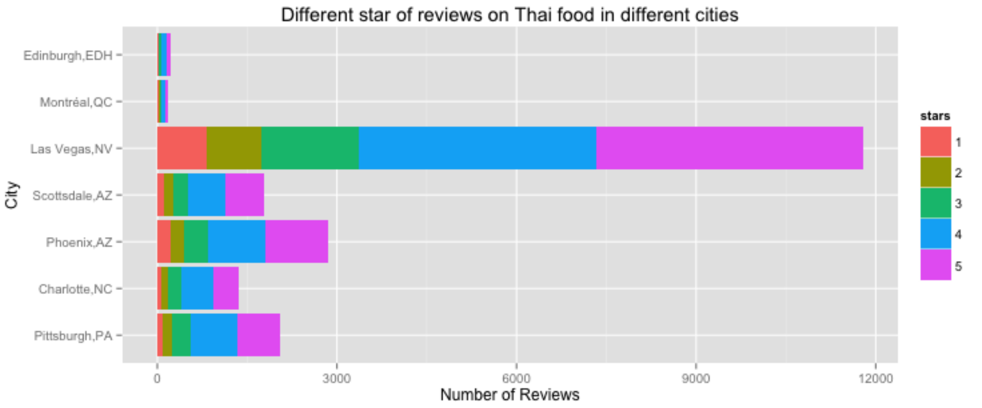


Figure 3.3: Stars of reviews about Thai food

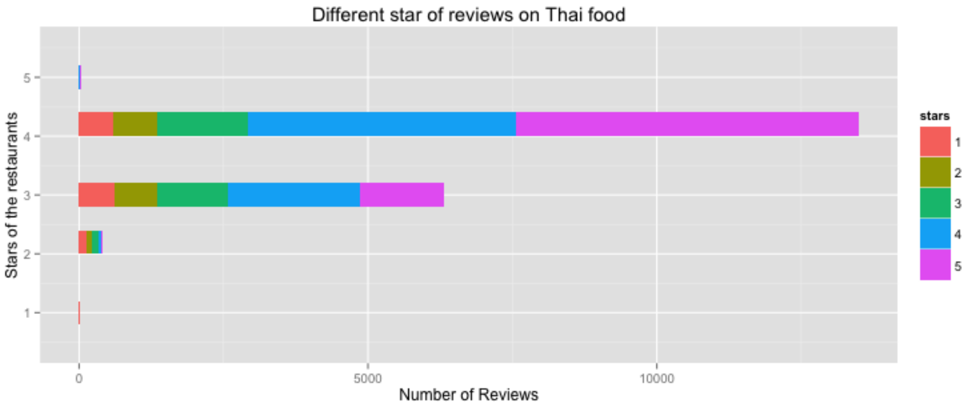


From 3.5, we can see that the users' rating stars in Phoenix is not normally distributed but rather left-skewed which indicates that most users are give pretty high stars by others, especially around 4. It shows that most users are being responsible for their views and are recognized by others. From 3.6, the x-axis is the number of months since Yelp launched its business on January 2004. The plot shows that during its first seven years, the number of users increases exponentially from 0 to around 15000. After seven years, the increased user number of every month decreased a little but still at a very high level until now.

Since the number of users is quite large, it is not plausible to compare every user's features to others, so a tractable way to deal with it is to cluster users to different groups. We used k-means to do the clustering and the features of users we used are listed as below:

- Number of reviews

Figure 3.4: Stars of reviews about Thai food



stars dist.png

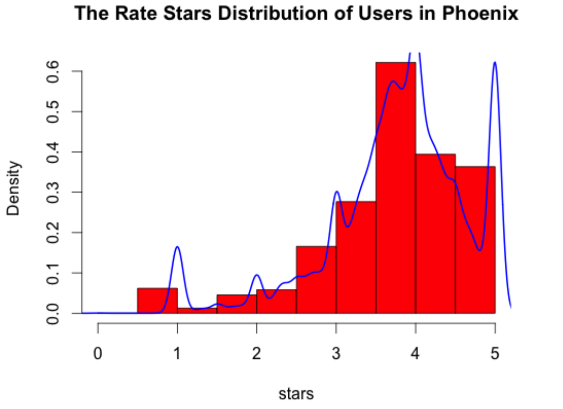


Figure 3.5: The Rate Stars Distribution of Users in Phoenix

used time.png

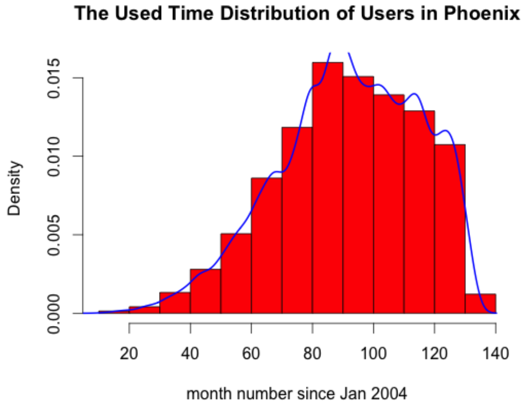
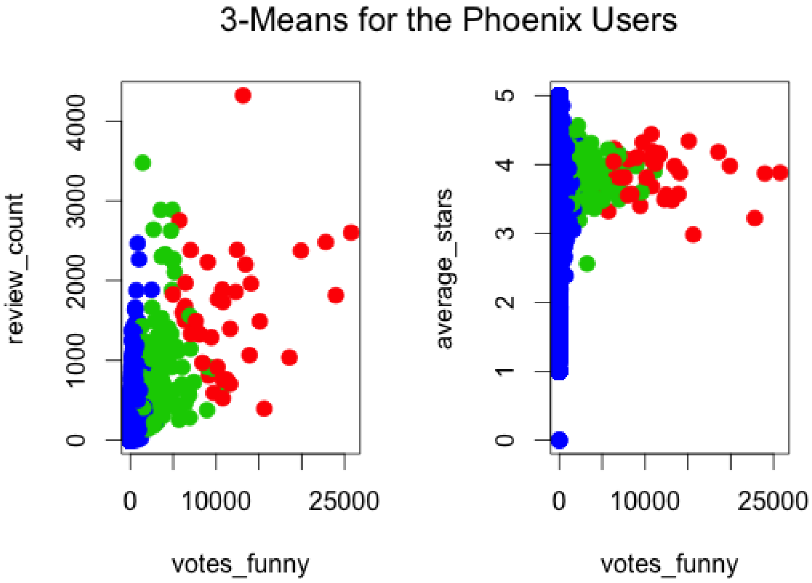


Figure 3.6: The Used Time Distribution of Users in Phoenix

- Average number of stars
- Number of "Funny" votes
- Number of "Usefu" votes
- Number of "Cool" votes
- Date of starting Yelp

k-means clustering aims to partition 45867 users into 3 clusters in which each users belongs to the cluster with the nearest mean. After applying the k-means, we found that the 3 clusters have its own feature. We can attach labels with each cluster. The first cluster contains 45633 users who tend to be least active such as has fewer votes and number of reviews; The second cluster contains 196 users who tend to be medium active, they all have around thousands of votes and hundreds of reviews; The third cluster contains 38 users who tend to be most active and their review may have much more influence than other users. We can see some more details from figures below:

Figure 3.7: K-Means Clustering for the Phoenix Users (K=3)



The plots tell that number of votes_funny and number of reviews have the positive correlation. And for the average stars of user, it shows that if users have very few votes for funny, it is hard to tell his or her rating stars. But if users have more than a hundred votes for funny, it tend to have high star levels.

3.3 Business Star Prediction

In the data set, there are 2653 restaurants in Phoenix and 141200 reviews about restaurants there. Based on this subset, we try to predict a restaurant's star level. We choose 15 variables as predictors. They are mean value of review stars there, longitude, latitude, number of reviews, whether can smoking, whether has wifi, noise level, whether can take reservations, delivery, parking, whether has TV, attire style, waiter service and alcohol.

First, try to find the relationship between means of review stars and restaurants' stars. Among 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, we find the nearest point of mean value of review stars to predict a restaurant' star level. We can correctly predict 2147 (80.9 percent) restaurants' star level. If the allowable error range is 0.5, the accuracy rate is 0.99. There are 6 observations which have residuals larger than 1. All of these 6 restaurants have few reviews(less than 8). It is clear that if a restaurant have enough reviews, we can predict restaurant's star level by mean value of review star level.

Then, try to find relationship between restaurant's star level and other predictors. We get a subset of 168 restaurants which do not have NA in these predictors and employed glm function in R. The classification performance is not so good as using mean value of review stars. However, we can find that the first three most important predictors except reviews are parking, alcohol, and noise level.

3.4 Restuarant Recommendation

After doing some basic analysis of users in Phoenix, we want to predict and recommend new restaurants to the users. Knowing about the business, the user's taste and others' previous experience about that restaurants are important. We can recommend restaurants according to either the stars of users given to the restuarant or how often the users went. We decided to use both way to get the recommend restuarants for users and put them together as the final recommendation restuarants. The flow chat of this process is shown on the right of the page:

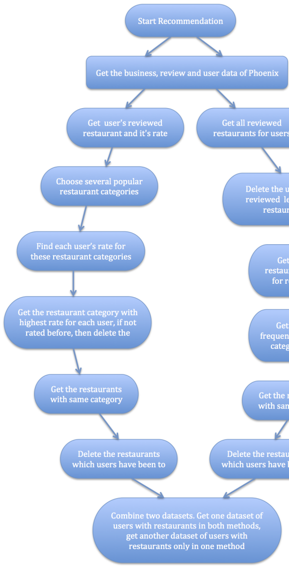

Start Recommendation
Get the business, review and user data of Phoenix
Get user's reviewed restaurant and it's rate
Get all reviewed restaurants for users
Choose several popular restaurant categories
Delete the users who reviewed less than 5 restaurants
Find each user's rate for these restaurant categories
Get all reviewed restaurants' categories for remained users
Get the restaurant category with highest rate for each user, if not rated before, then delete the
Get the highest frequency of restaurant category for users
Get the restaurants with same category
Get the restaurants with same category
Delete the restaurants which users have been to
Delete the restaurants which users have been to
Combine two datasets. Get one dataset of users with restaurants in both methods, get another dataset of users with restaurants only in one method

chart.png

Figure 3.8: Flow Chart for Restuarant Recommendation

From the flow chat, we can see that for the method using the stars of restuarant given by users, first we got several categories of restuarant that users often go to, then for each categoy we find the every stars which each user gave, then found category with the highest stars for each user to be the restuarant category we will recommend later. The other method is to find the category of restuarant that each user most went to before, then choose this category as the recommend label. The results show that we did restuarants recommendation only for 26869 out of 45867 users. Among these users, there are 2867 users whom we used two methods to get the intersection of recommended restuarants for, this means they may get more appropriate recommended restuarants and more likely to go and have a try. The other 24002 users get the recommended restuarants from either methods which seems less reliable than the previous one, but it still worth to try. The recommended restuarant business_id for each users in Phoenix is in the recommed.Rda dataset which can be found from Bitbucket[2].

4 Features Generation

In this section we start with describing simple data manipulation to create user dependent and location dependent features. Then we use opinion extraction from metadata review text to caste usable features vectors.

4.1 Location and User Dependent Features

Here we have two categories of features. The first category contains features which describe metadata about business locations, including longtitude and latitude, whether the restaurant is family friendly or has wifi, the number of business within 2 miles radius that share a category. The second categories mainly carry features about users' taste.

The following list describes as features:

- Number of reviews in a set
- Average stars across reviews in the set
- Location of the business and its attributes
- Number of checkin users since the first review
- Business type (Different type of subcategories under Restaurant)

We expect features describing the location of the business to be userful for calculating attention of restaurants and predicting future reviews. Therefore, we generate multiple subsets that contains all reviews for restaurants located in the most popular cities.

4.2 Quality Phrase Mining

This section we build a corpora of 720K reviews, consisting of over 93 million words written by 30k reviewers using Python and NLTK toolkit. Our goal is to mine most frequently occuring informative keywords among all restaurant reviews and analyze the sentiments for each of these keywords and generate related features. We first present the full procedure of the keyword and phrase mining and then we will introduce each of them in following subsections. 1. Generate a list of frequent word and phrase candidates (top 500 key words) and estimate phrase/keyword quality based on informativeness requirements and its correlation with other keywords. (See 3.1.1) 2. Categorize the sentiments critiquting those top keywords for each review. (See 3.1.2)

[2] The Bitbucket URL: *git@bitbucket.org:Yuki0425/sta242_project.git*

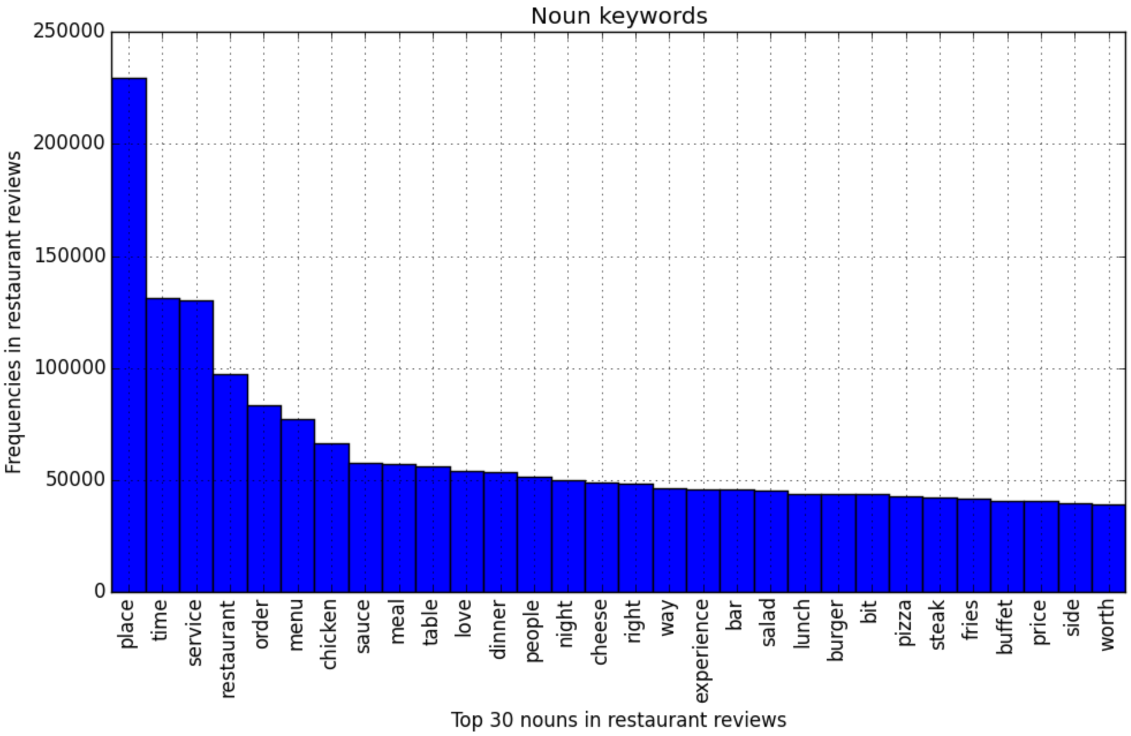
4.2.1 Word Sequence Segmentation and Parsing

Quality phrases should occur with sufficient frequency in a given document collection. To generate a list of popular words, the first thing is cleaning up the text such as removing excessive punctutation, spell checks, removing unnecesary emojis and so on. Then we run the Yelp copora through a series of tokenization - we select ten most popular cities ad tokenize each user provided review into sentences, then tokenize the sentences into words. We suspect that a considerable portion of the *word tockens* in reviews are function words, contentless words, and punctuation. We only want to keep the words that are informative and indicative of a specific topic. "This place" might be a popular and concordant phrase, but might not be informative in research our Yelp corpus. Therefore, it is necessary to cull the not so informative words before we calculate the frequency distribution of *word tokens*. After cleanning up the raw reviews, we then part-of-speech tag each token in every sentence of the user reviews. Due to the fact that the metadata of the reviews are enormous, we consider apply probabilistic tagging method using the Penn TreeBank training corpus. Probabilistic tagging assigns a tag to a word based on a probability that such word is of the target category. In this case, the probabilities are calculated ahead of time, with help of NLTK TreeBank corpus. Therefore, we use a sequential unigram tagger built on the training corpus to assign the most likely tag for a token based on the highest probability for that tag in the TreeBank corpus. Most importantly, the sequential tagging tags are assigned to one token at a time, beginning with the initial token and moving on sequentially through the following tokens in the next, therefore reducing the risk of miscategorizing ambiguous words in phrases and word sequence. Below is an example of a parsed sentence ("This used to be our go to breakfast when visiting Phenoix but we swear things have changed recently.") from a random selected sentence from a restaurant review in Pheonix, AZ.

```
[(’This’, ’DT’), (’used’, ’VBN’), (’to’, ’TO’), (’be’, ’VB’),
(’my’, ’PRP$’), (’go’, ’VB’), (’to’, ’TO’), (’breakfast’, ’NN’),
 (’when’, ’WRB’), (’visiting’, ’VBG’), (’Phoenix’, ’NNP’),
  (’but’, ’CC’), (’I’, ’PRP’), (’swear’, None), (’things’, ’NNS’),
  (’have’, ’VBP’), (’changed’, ’VBN’), (’recently’, ’RB’)]
```

For the purpose of this project, We care only about nouns (labeled `’NNS’, ’NNP’, ’NN’`) and adjectives (labeled `’JJ’`). One tricky part worth pointing out is that it is necessary to perform text chunking to create relevant noun phrases in order to avoid casting a conditional independence assumption on the words. A noun phrase is a phrase which has a noun as its head word and performs the same grammatical function as such a complete phrase. And a complete phrase should be interpreted as a whole semantic unit in certain context. Typical examples of a properly chunked phrase could be "two tasty homemade pies". But We am not ver concerned with grouping adjective that followed after the first noun. Rather, We am more interested in the text chunks that two or more nouns appear together, for examples, ’Pad Thai’ and ’spring rolls’. We define a grammar function that

Figure 4.1: Keyword (nouns) generation



only chunks nouns that appear together, therefore in that cases above, 'spring' and 'rolls' will be grouping together into a userable noun phrase. This method allows me to avoid making the strong conditional independence assumption as Naive Bayes when we select most popular keywords.

After running the above computation over all sentences for all restaurants in Vegas, Pheonix, Charlotte, Pittsburgh, and Montreal while keeping counters of all noun phrases, we calculate the frequency distribution of all the words and obtain a dataset with 1000 most common words. At the end, we take a subset of the keywords, around 120 words, that appeared the most among all the restaurants' reviews and use those to generate feature vectors given a single review. The plot below is a frequency histogram of the top 30 noun keywords. The reason we only choose to generate vectors across restaurant cagetory is because that running the above algorithm across many different categories might add unrelated keywords to the final set of keywords. For example, keyword "oil change" seems irrelevant to "losbster buffet".

The plot below shows that the most frequently occured nouns are words such as "place", "time", "service", "order", "menu" and directly followed by more informative and descriptive nouns like "chicken", "sauce", "cheese", "bar", "pizza", "burger", "buffet" and so on.

4.2.2 Opinion Extraction and Sentimental Analysis

With a representative set of keywords (food type and other nouns) to describe restaurants, we can now extract users' opinion towards an individual restaurant by counting the number of positive and negative opinions about each keyword.

First of all, we generate a set of positive adjectives and a set of negative adjectives use Wordnet as described in [1]. Wordnet is a lexical database for the English language and groups words together into sets of synonyms and antonyms. we select some popular adjectives from user reviews (see algorithms in previous section) and then select the positive ones accoding to the polartiy scores using SentiWordNet[3]. Starting with a small list of positive seed adjectives (polarity score >1), We accumulate all synonymous adjectives to the seed and use those as our positive words dictionary. Similary, we obtain our negative words dictionary by accumulating all antonyms. With a list of positive and negative adjectives and a list of keywords of nouns and noun phrases, we am able to analyze each sentence in each review by counting the number of positive and negative adjectives associated with the keywords [2]. More books and papers regarding to the methodology of opinion mining can be found at the website of Hu and Liu's lexicon research in University of Illinois at Chicago[4]. For the sentence "Ok, Don't have their grilled egg rolls or crabpuffs, they'are soggy and floppy and just so gross", Assuming 'egg rolls' is our keyword, then we select out the adjectives (`JJ`) in the sentence and check if they are in our positive or negative dictionary. For this case, since *'gross'* is associated with 'eggrolls' and it's in our negative adjective dictionary, this sentence would have one negative count for the keyword "eggroll'. For sentences with multiple noun phrases and adjectives in them, we associate the adjective with the closest noun phrase.

By mining direct opinions over all reviews for restaurants in Las Vegas, Pheonix, Montreal, Pittsburgh, and Charlotte, we hope to extract good and bad features of one selected business. Presumably, if the restaurant has the best *eggrolls* in Pheonix, we would collect many positive counts and only a few or none negative counts about that keyword in the reviews.

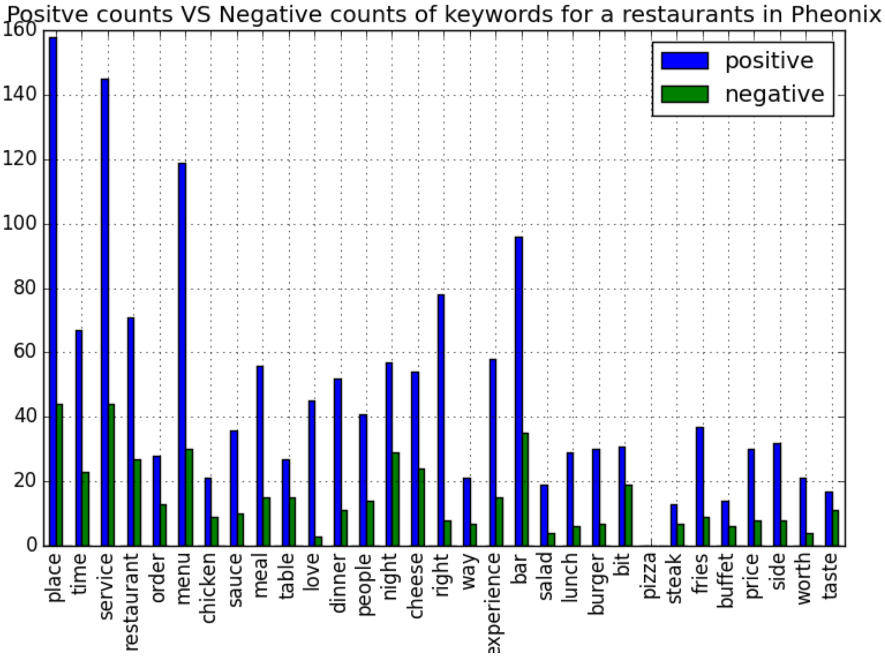
Below shows a slice of positive and negative opinion counts on thirty keywords for one quite popular restaurant in Pheonix:

It is pretty obvious that from the graph the selected restaurant is favored by most of the users and seems like that restaurant has amazing menu at night with a nice bar and good service, and the restaurant has tasty cheese.

[3] SentiWordNet is a publicly available lexicon. The resource contains Princeton WordNet data marked with polarity scores. It can downloaded from: *http://sentiwordnet.isti.cnr.it/*

[4] Opinion Mining, Sentiment Analysis, and Opinion Spam Detection: *http://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html*

Figure 4.2: Example of Opinion Extraction for one restaurant



5 Feature Selection and Conclusion

For predicting and recommending new restaurants to a user, knowing about the business, the user's taste and others' previous experience about that restaurants are important. This projects explains the process of extracting features through data manipulation and keyword-opinion mining. We use 15 features to generate a regression model to predict a restaurant's star rating. Also, we come up with a method to recommend new restaurant to user based on the category of the new restaurant and user's preference. The next step will be to combine the positive and negative opinion (from other users) on certain keywords into our recommending process. However, it is not clear that whether the extracted features contain redundant or irrelevant information when we combine them together. So it might be a good idea to use a naive approach to perform feather selection using an exhaustive search over all possible subsets of features and pick the subset with the smallet test error. Moreover, use K-Fold cross-validation can also minimize test error.

References

[1] Bing Liu, Minqing Hu and Junsheng Cheng. "Opinion Observer: Analyzing and Comparing Opinions on the Web." Proceedings of the 14th International World Wide Web conference (WWW-2005), May 10-14, 2005, Chiba, Japan.

[2] Minqing Hu and Bing Liu. "Mining and Summarizing Customer Reviews." Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), Aug 22-25, 2004, Seattle, Washington, USA,