YELP ANALYTICS

BY

AAYUSH AGRAWAL

A thesis submitted to the
Graduate School—Camden
Rutgers, The State University of New Jersey
in partial fulfillment of the requirements
for the degree of
Master of Science
Graduate Program in Scientific Computing

Written under the direction of
Dr. Sunil Shende
and approved by

Dr. Sunil Shende

Dr. Jean-Camille Birget

Dr. Suneeta Ramaswami

Camden, New Jersey

January 2017

THESIS ABSTRACT

Yelp Analytics

by AAYUSH AGRAWAL

Thesis Director:

Dr. Sunil Shende

Yelp is a website and mobile app which publishes crowd-sourced reviews about local businesses. In this thesis, we analyze data about restaurants from Yelp, specifically the reviews, to predict the star-ratings of the restaurants based on the contents of the reviews. Our results are based on performing sentiment analysis on the reviews, which involves determining whether a review is positive or negative. Various machine learning techniques were applied to the data after appropriate extraction of linguistic features, to create classification models, and to predict star—ratings based on these models.

Acknowledgements

I would like to express my sincere appreciation to Dr. Sunil Shende for his guidance on this project.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Yelp is a website and mobile app that connects people with local businesses. It is a guide for word-of-mouth on everything from boutiques and mechanics to restaurants and dentists. The Yelp community is made up of engaged locals who connect on-line and off-line to share their opinions about local businesses. Millions of users connect on this application to, rate and give reviews for businesses and thus, there is an interest in performing sentiment analysis of the reviews.

However, deriving the sentiment of a review using a machine is not a trivial task. A rule—based sentiment analysis method, is not effective for all cases and will not give good results. Sentiment is a result of not only the presence of some words (having a positive, negative or neutral sentiment) being used in a review, but also the "stance of a writer". For example, a review might look positive, but may be a sarcastic review. To identify sarcasm we could use the tone of voice (which is not available to us in the text) so combinations of terms within a review which may not be contiguous to each other can contribute systematically to the total sentiment of a review.

For example, previous work in sentiment analysis (by Mohammad et al.) [6] created two state-of-the-art SVM classifiers, one to detect the sentiment of messages such as tweets and SMS (message-level task) and one to detect the sentiment of a term within a message (term-level task). The National Research Council (NRC) is the Government of Canada's premier research and technology organization (RTO). Sentiment of a review can either be positive, negative or neutral. SVM is a classification technique in Machine Learning, which takes some reviews to start with thats already classified into positive and negative (the training

set), and tries to predict a set of unclassified data (the testing set). Classification was done for the probability of a review to be positive or negative. They used macro-averaged F-score to measure the performance of the classifier. They attained F-score of 69.02 in the message-level task and 88.93 in the term-level task. F-score is a measure of accuracy. It uses precision & recall to compute the score. Precision is a measure of result relevancy, while recall is a measure of how many truly relevant results are returned. Macro average is simply a mean for each class, that is positive and negative sentiment.

NRC participated in an international competition organized by the Conference on Semantic Evaluation Exercises (SemEval-2013). The organizers created and shared sentiment-labeled tweets for training(8,258 tweets), development(1,654 tweets), and testing data(3,813 tweets). Another dataset was given for SMS messages separately (no training data for the messages was given separately and the model trained on the labelled tweets, i.e. the training data was only used for the classification of unlabelled data). Sentiment lexicons were used which are collections of words along with associated positive or negative sentiment.

Two classes of sentiment lexicons were used.

- Manual (Already existing) Lexicon: NRC Emotion Lexicon (Mohammad and Turney,2010; Mohammad and Yang, 2011)(about 14,000words), the MPQA Lexicon (Wilson et al., 2005)(about 8,000 words), and the Bing Liu Lexicon (Hu and Liu,2004) (about 6,800 words)
- Automatically generated lexicon : The hashtag of a tweet, words such as joy, sadness, angry, and surprised are good indicators that the tweet as a whole is expressing the same emotion as the hash tag. They used these hashtag's to get positive and negative tweets. These terms were chosen from entries for positive and negative in the Rogets Thesaurus.

These tweets were then used to generate a large word sentiment association lexicon. The hash tags are considered the pseudo labels of a tweet and a score of each term in a tweet is calculated. This lexicon was generated for unigram, bigram, unigram pairs, bigram pairs & (unigram,bigram) pairs.

Once the sentiment lexicons were made ready, the features for the SVM classifier were developed. Each feature was composed of a group of features: word ngrams (presence or absence of contiguous sequences of 1, 2, 3, and 4 tokens; non—contiguous ngrams), character ngrams (presence of absence of 3,4,5 contiguous sequences of characters), all-caps (the number of words with all characters in upper case), POS (frequency of POS tags), hash tags (frequency of hash tags) and some other features were used. Each review was represented as a collection of these features and then used in training set (8,258 tweets). The development(1,654 tweets) and Testing set(3,813 tweets) were used to test the trained model on the unannotated data (unseen data).

Emoticons like $\{:), : (, : D..\}$ were tokenized using Christopher Pott's tokenizing script[5].

In our thesis we computed TF-IDF (term frequency-inverse document frequency) matrix for all the unigrams, bigrams, and all pairs of (unigram,bigram), (unigram,unigram), (bigram,bigram). These TF-IDF matrices were used to build automatically generated sentiment lexicon. All other features were used in the same manner in our thesis as described above to build the model for predicting the sentiment of reviews. We are predicting the probability of a review to be having a positive or negative sentiment.

In our paper, the main objective is to use the provided data to derive interesting insights about businesses and reviewers. Reviewers for these businesses will be referred to as Yelper's from here on.

1.1 Outline

1.1.1 Exploratory Data Analysis

In Chapter 3 we studied the distribution of businesses in the city of Pittsburgh, USA and derived conclusions about their distribution from their location (latitude, longitude) information. We did K-means clustering of businesses in the city of Pittsburgh based on this location attributes. Clustering is a technique for finding similarity groups in a data, called clusters. k-means is an unsupervised learning algorithm for clustering. Each

business category (Fast Food, Restaurants) or (Nightlife) etc.) is composed of sub-categories separated by commas. The basic motivation behind dividing a category into subcategories was to find out which set of sub-categories defines the nature of a cluster. We compare the frequency of occurrence of each sub category inside each cluster, and top 5 are selected to be the nature of the cluster. Also we generated heat maps of 1 and 5 star rated businesses in United states which showed us the distribution of stars.

We identified users who had done the highest number of reviews for each category. The category for a business will be a collection of strings such as (Sushi, Restaurant) which uniquely identifies what a business is actually about. We found user's with maximum number of reviews for each category for businesses - and called them as trend setters. We identified the trend setters only for the most popular and least popular businesses. We measured popularity for a business by its Annual growth rate metric, which is dependent on the number of reviews done for the business per year. It is a percentage value. A high value of growth rate for business indicates it is very popular, and thus it would be interesting to know about users who are actually making them popular by writing the highest number of reviews for that business category. On the other hand, if a business is not doing so well, by which we mean that not many people are talking about it on Yelp then it would be interesting to see those users who are setting the negative trend for these least popular businesses.

1.1.2 Tagging and categorization of Yelpers by Knowledge

In section 3.3.4 we classified each reviewer as knowledgeable or not knowledgeable about a business. For each business we extracted all reviews and the users who gave those reviews. For each user we checked the friend connections. If we found at least one friend of a user who has also given a review for the same category of business as the user has, then that user is marked as knowledgeable. Users who have no connections to friends who have given at least one review in the same category of the business will be marked as having no knowledge.

1.1.3 Natural Language Processing for sentiment analysis of reviews

In Chapter 4 we performed a review level sentiment analysis. Section 4.2.2 discussed about the technique used to build the Automatically generated lexicons. We implemented a number of features for the model discussed in section 4.3 and trained Linear, SGD, Elastic-Net,Ridge regression models to do sentiment analysis for the reviews as a whole. We implemented a variety of features based upon the sentiment lexicon. Using these models we were able to predict the star rating of each review in the test data and used mean squared error to measure each model. Results of our sentiment analysis experiment are discussed in section 4.4.

Chapter 2

Description of Data

The data set provided by Yelp, contains data from the following countries, for users and businesses in the cities listed in parentheses: United Kingdom (Edinburgh), Germany (Karlsruhe), Canada (Montreal and Waterloo) and United States (Pittsburgh, Charlotte, Urbana-Champaign, Phoenix, Las Vegas and Madison).

We used three specific data files:

1. `business.json`: Business metrics like address, working hours etc.
2. `review.json`: Details about the user, review text, date etc.
3. `user.json`: User information like name, average number of stars given to businesses in a review etc.

These files were converted in CSV format before loading into Python dataframes for analysis in Jupyter notebooks. Given the large size of the `review.json` file, we adopted the process of chunking data from the file in batches of 100000 records each.

2.1 Data objects

- For the purpose of analysis, we only selected a subset of relevant attributes for each data object; these attributes are detailed below.
- Each data object has at least one attribute which serves as a *primary key*, e.g. Business_ID, User_ID, Photo_ID.

2.2 Review

1. The review data object contains information on the reviews for businesses submitted by Yelp users (henceforth called Yelpers).

2. The "stars" field is a numerical rating given by a Yelper to a business. It is rounded to the nearest half point. The "text" field contains the actual review written by the Yelper. The "date" field provides the date on which the review was written by the user and the "votes" field is a count for each category of votes.

3. Keys: Business_ID, User_ID

4. An example of a review is given below:

Table 2.1: A Review Record

Column Name	**Value**
`business id`	5UmKMjUEUNdYWqANhGckJw
`Date`	2014-02-13
`Stars`	5
`Text`	Excellent food. Superb customer service
`User_ID`	Iu6AxdBYGR4A0wspR9BYHA
`Votes`	'useful': 0, 'funny': 0, 'cool': 0

2.3 User

1. The user dataframe holds information such as the average number of stars given for businesses by the user.

2. Compliments are given by other Yelpers for a user, while Elite is a status given to certain Yelpers (in certain years) which indicate more trust to their reviews. Elite yelpers get the opportunity to attend exclusive events, meet yelpers from the same community in-person, and discover a variety of local businesses that may not have been previously tried.

3. Keys : User_ID

4. An example of a user is given below:

Table 2.2: A User Record

Average_stars	4.14
Compliments	'note': 20, 'hot': 48, 'writer':9
Elite	2005, 2006
Fans	70
Friends	-6rEfobYjMxpUWLNxszaxQ
Review count	108
Name	Russel
User_ID	18kPq7GPye-YQ3LyKyAZPw
Votes	'useful': 280, 'cool': 245, 'funny': 167
Yelping_since	2004-10

2.4 Business

1. Keys : Business_ID

2. An example of a business is given below:

Table 2.3: A Business Record

Attributes	'Good for Kids': True
Business_ID	cE27W9VPgO88Qxe4ol6y_g
Categories	Active Life, Mini Golf, Golf
Full Address	1530 Hamilton Rd Bethel Park, PA 15234
Latitude	40.354115
Longitude	-80.014660
Name	Cool Springs Golf Center
Stars	2.5

Chapter 3

Exploratory Data Analysis

Yelp data is rich in detail including reviews, locations of businesses being reviewed etc. We performed some exploratory analysis on the data for Pittsburgh to identify interesting patterns. By doing k-means clustering analysis of the locations for the businesses in Pittsburgh, we were able to categorize businesses geographically as well as in terms of categories like 'Fast Food', 'Restaurants', 'Nightlife' etc. Our exploratory analysis shows interesting patterns about that the nature of clusters, e.g. car towing businesses are distributed uniformly around the city.

A *category* for a business is a collection of strings which summarizes the *nature of the business*, for example : ['Fast Food', 'Restaurants']. A category may be subdivided further into sub-categories. We do this by considering each category as a string and then breaking it into parts containing different sub—strings, since each subcategory is separated from each other inside a category by a "', '". We then parsed the sub—strings to remove the characters which are not relevant to a sub—category like : "[", "]", "'". Example of a business category ('Food', 'Grocery', 'Mexican', 'Restaurants') will be broken down into the sub—categories : 'Food', 'Grocery', 'Mexican' & 'Restaurants'.

3.1 Points pattern analysis & feature map

Recall that businesses are tagged with location information: their latitude and longitude. We studied the distribution of business locations in the city of Pittsburgh, USA, to obtain geographic and visual information about businesses alongside their categories: we did this by *mapping* businesses by latitude and longitude information on geographical maps (Google

maps) after appropriate filtering by categories. By filtering the features of each business (Categories, Stars etc.), it is possible to get visual information about the distribution of businesses with certain characteristics in some geographic area.

For example, we used strings like 'Fast Food' to search and identify categories like ('Fast Food', 'Restaurants') etc. For each category, we computed the the sum of distances (in kms.) between all pairs of businesses in that category, which allowed us to calculate summary statistics like the average distance between all the business pairs in a category, the average-latitude/longitude for businesses in a category, and the density of businesses in a category.

- Descriptive Spatial Statistics: Using the latitude and longitude information, we calculated the centroid location for businesses in a category. The centroid minimizes distances to all the points (businesses) in the category. For example, we learnt that at the point 40.46 Degrees North and 79.96 Degrees West is the centroid of all pizza businesses as seen in Fig 3.1.

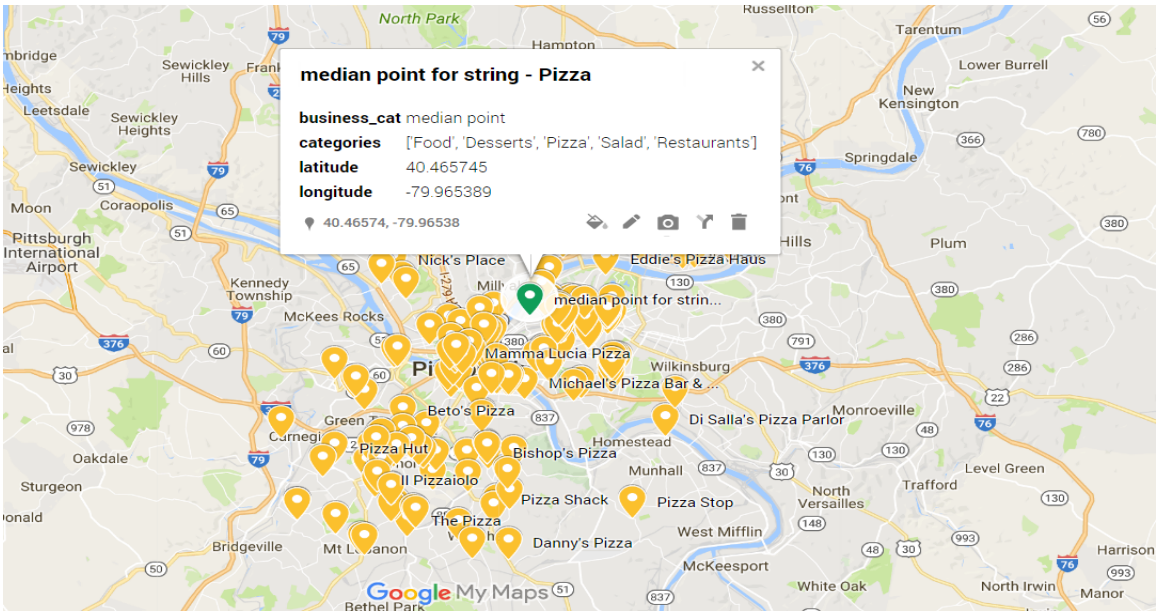

Figure 3.1: Feature map of Pizza businesses and their median point in Pittsburgh

Similar analysis and visualization for pubs (Fig 3.2), car towing businesses (Fig 3.3) and italian businesses (Fig 3.4) in Pittsburgh shows interesting patterns.

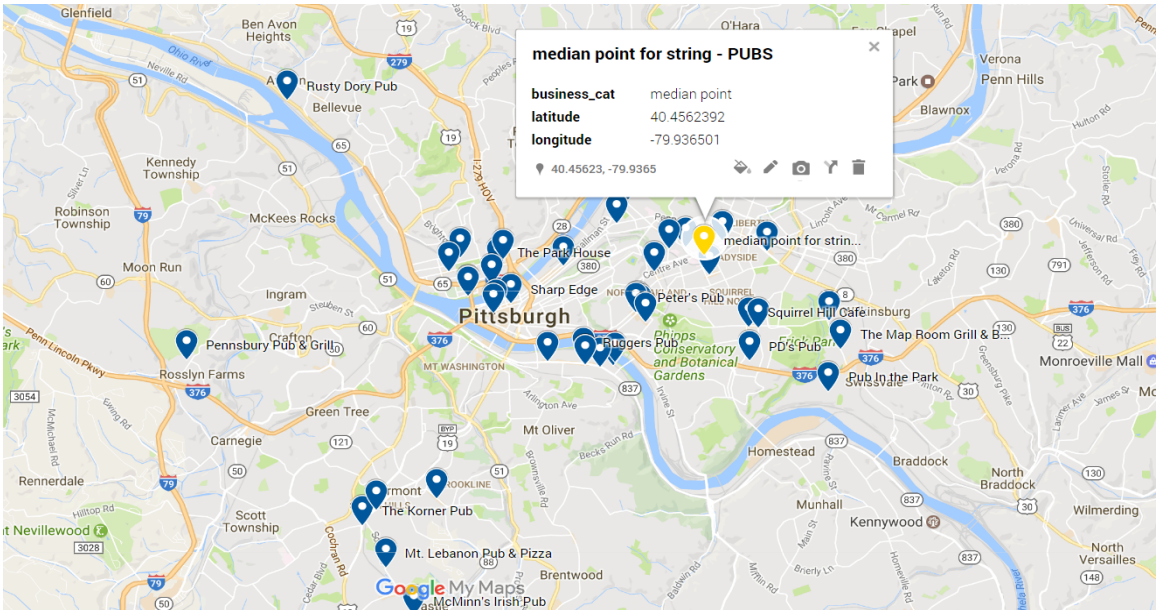

Figure 3.2: Feature map of Pub businesses and their median point in Pittsburgh

- K-means clustering of businesses: Using the K-means clustering algorithm, we partition businesses geographically into several clusters.

  The elbow method, which allows us to determine an appropriate number of clusters, was used to group businesses. For example, as shown below in Figure 3.5, we choose 4 clusters for the k-means clustering algorithm when applied to data for Pittsburgh.

  The basic motivation behind dividing a category into subcategories was to determine the correlation among geographic cluster information and the subcategories identified: for instance, the frequency of occurrence of each subcategory inside a cluster, and using the top 5 subcategories to give a qualitative characterization of the nature of the cluster. This cluster-wise data is used to populate a scatter plot along with a defined legend about the nature of each cluster. Looking at the map we can know which cluster of 'Pittsburgh' is composed of which major subcategories.

- A representative kind of feature analysis is seen in Figure 3.9, where we mapped businesses in the city 'Apache Junction' with category 'Automotive Towing'. Note how with respect to the 3-way junction, towing businesses are located in such a way that they cover distinct spokes emanating from the junction and thus are in a position to maximize their revenue over the section they cover. If all three would have been

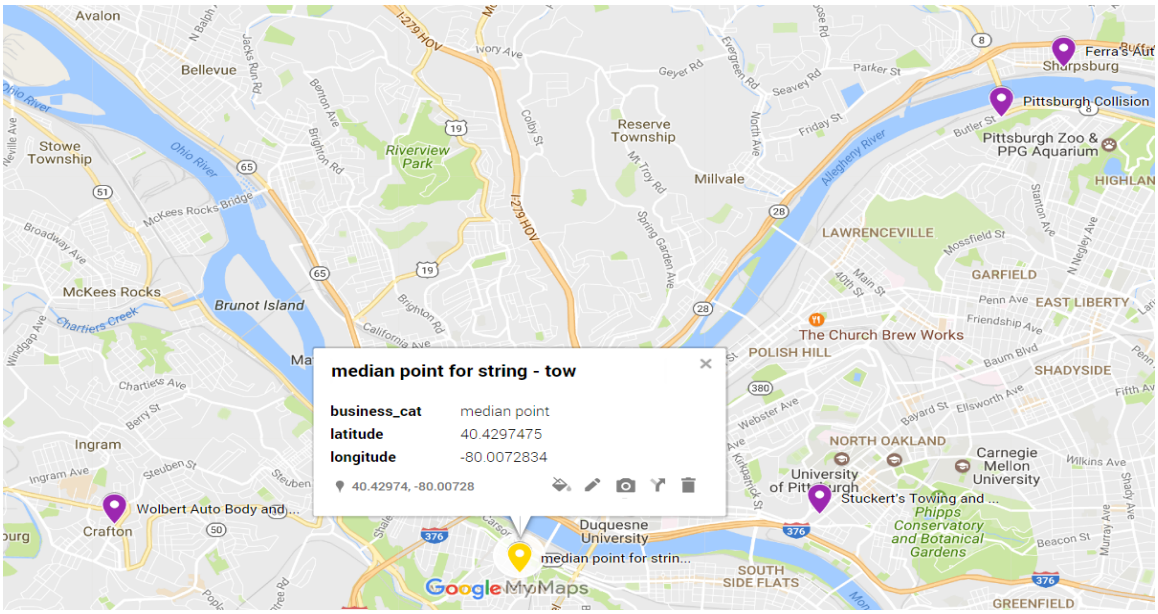

Figure 3.3: Feature map of Car Towing businesses and their median point in Pittsburgh

at the same location or near by then that would have caused loss to each one due to intense competition.

- Another representative visualization is seen in Figure 3.10 that depicts businesses in the city of Champaign under the categories 'Indian' and 'Restaurants'. All three businesses shown have star ratings of 3.5, which is an average but not exceptional rating for a restaurant. Upon exploring the attributes for each of these 3 restaurants separately, we found that there were some desirable attributes for restaurants that were absent in all the three places such as Outdoor Seating, Alcohol, Good for Kids but not romantic ambiance, not open late night etc. This kind of information further points to the relevance of linguistic terms in reviews being correlated with ratings.

3.2 Heat-map based on geography

- Introduction A Heat map based on the geography is used to show patterns which otherwise may be difficult to detect or derive from the data. Patterns which we are interested are in the distribution of businesses of a particular category (fast food, chinese etc.), stars(1-5) in and around a location on a map. These patterns can be

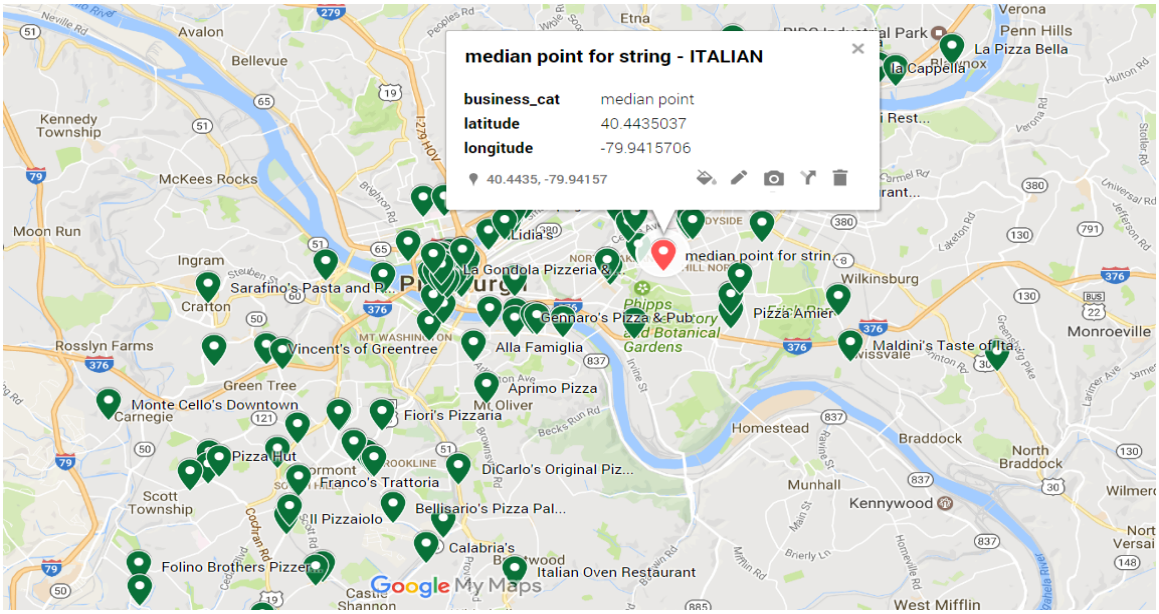

Figure 3.4: Feature map of Italian businesses and their median point in Pittsburgh

useful to understand if there is potential for new business to come up in a location an example; like there is no highly rated (lets say 4.5 star) Chinese restaurant in an area. There may be only 2 star rated Chinese restaurants in a location which gives a hint that people in that area may not be happy with that business. This can then be used as a factor for further analysis of those business reviews to boost its revenue or improve its rating or justify for entry of another business of the same category in the location being examined.

- The heat map is a visual depiction of the density of businesses in a particular area of the map. With light green to red, with red depicting most dense area for the selected category, stars or city of business.
- For the selection: categories containing 'Afghan' we get 8 cities namely : Chandler, Dollard-des-Ormeaux, Gillbert, Las Vegas, Laval, Madison, Montreal and Phoenix. However when we check for only 4-5 star rated business Chandler and Madison don't have any highly rated Afghan restaurants.
- We observed that there is a high concentration of 5-star rated businesses in Charlotte and Pittsburgh, however 1-star are high in number around Las Vegas and Phoenix.

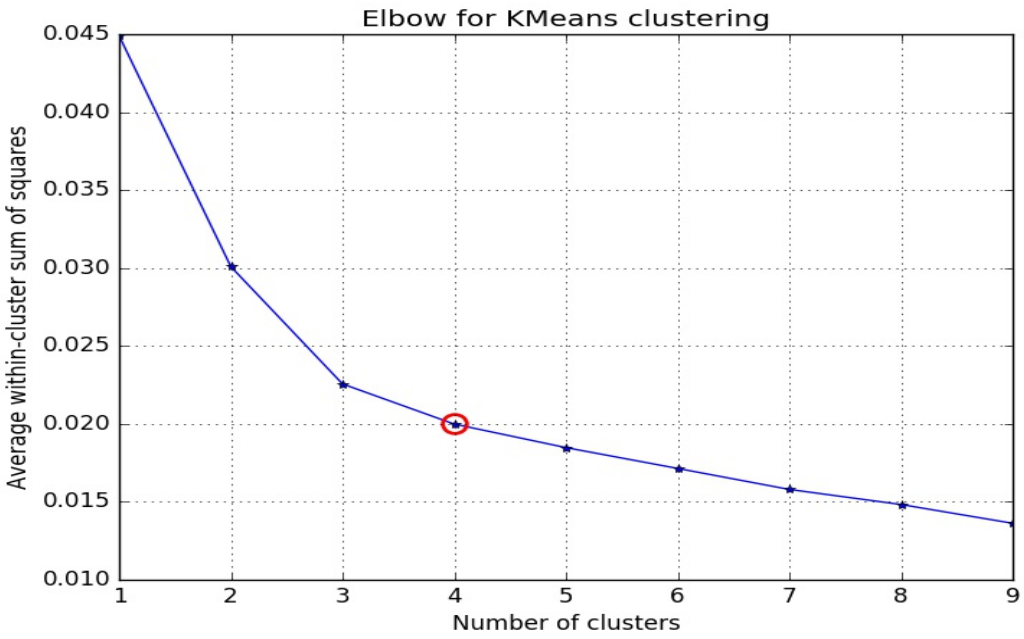

Figure 3.5: Elbow method to find number of clusters

3.3 Trend Setters

Our final representative exploratory analysis concerns *trend setters*: intuitively, these are users with the highest number of reviews for a category. Trend setters can influence businesses quite directly. We seek to identify them so that businesses can proactively react to negative sentiments in reviews to prevent loss of clientele or conversely, to positive sentiments in reviews to further improve performance in areas praised by trend setters.

Before identifying the trend setters or popular users for categories of businesses, we characterized businesses in terms of their popularity. We did this by finding the total number of reviews done in each year for a business. Then we calculated the annual growth over multiple years for each business.

We found the most recent year (final_year) for any review done & the total number of reviews done for year (final_num_reviews) for each business. Then we found the first year (start_year) which is the year on opening date of business for any review done by any user & the total number of reviews done for that year (start_num_reviews) for each business.

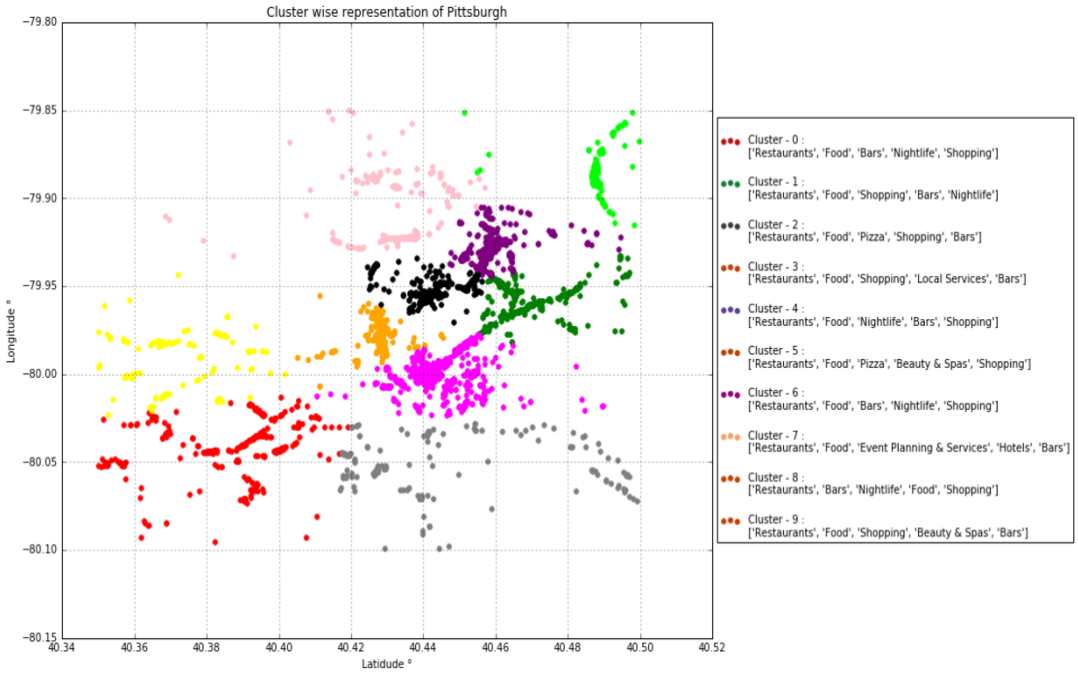

Figure 3.6: Scatter map of k-means clusters for the city - Pittsburgh, USA

Next we found the difference between the most recent year (final_year) & the first year (start_year), which gives us the number of years, b, for which a business has been reviewed.

We calculate the Annual Growth Rate for a business as follows:

$$AnnualGrowthRate = ((final\_num\_reviews/start\_num\_reviews)^{1/b} - 1) * 100$$

Then based upon annual growth rate and category, we find the top business and worst business for each category. We then find the trend–setters for these identified businesses by finding the user with maximum number of reviews done for each category.

Last, we take out each review done by the trend setters for these top businesses and do sentiment analysis on their reviews using the training model discussed in the next chapter on natural language processing for sentiment analysis of reviews. These reviews are particularly useful, because they are written by users who have done the maximum number of reviews for the category a business belongs to, and thus finding the sentiments in such reviews is crucial.

We found 6327 top businesses for which there were 6327 trend setters, and 5309 unique

Figure 3.7: Scatter map of 4 k-means clusters for the city - Pittsburgh, USA

trend setters (since a trend setter may be a top user for more than one category). We similarly found 3081 worst businesses for which there were 3081 trend setters, and 2579 unique trend setters.

3.4 Tagging and Categorization of Yelpers by Knowledge

The main idea here was to circumvent the issue that some businesses may falsely give high ratings to themselves by using popular yelpers to attract attention. These popular yelpers may be people with many friend connections or be elite users. The main idea is to shift the focus here from elite yelpers, to yelpers who are actually interested about the category. Elite members bear special icons on their Yelp profiles, they're invited to private events where up-and-coming restaurants and bars provide food and drinks to them for free. Any user can send a request to become a elite member, which are sent to the San Francisco-based Elite Council, a group that's responsible for making sure the applicant is a real person writing real, reasonable reviews of businesses. But according to Yelp, there isn't really a specific

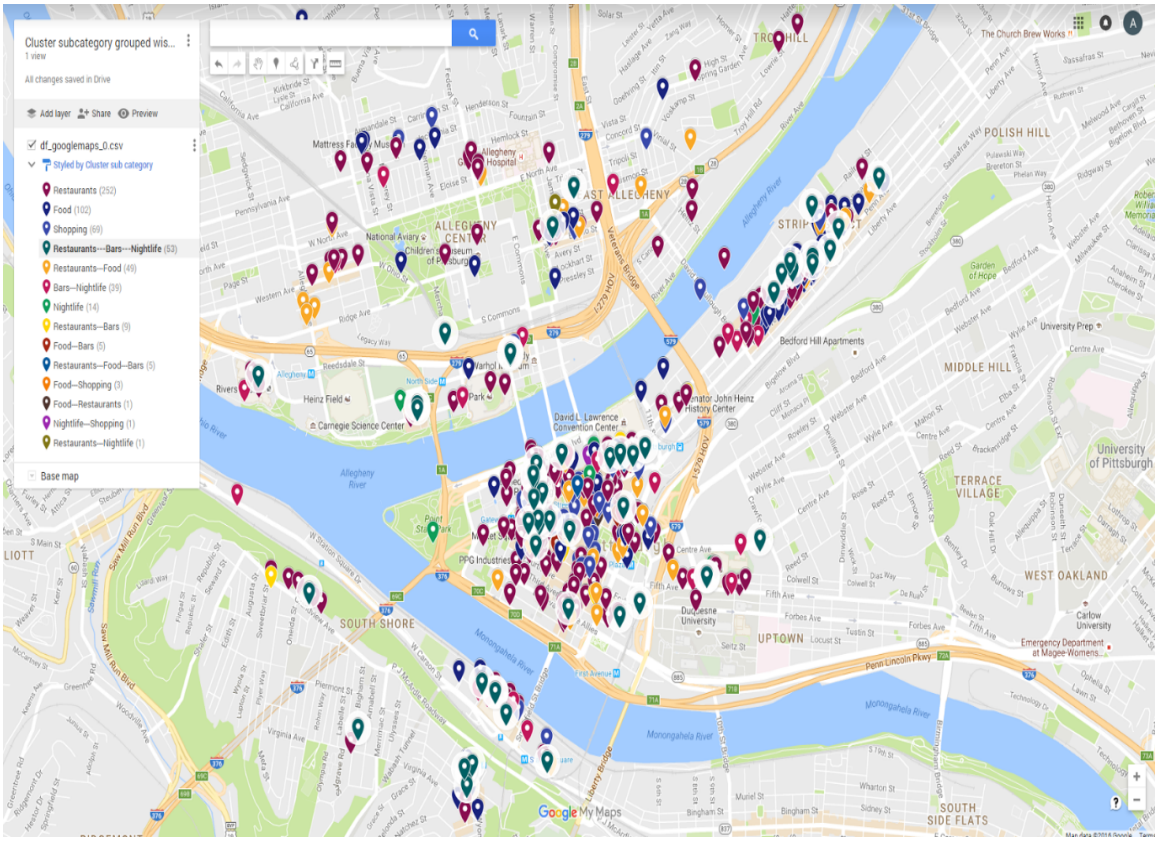

Figure 3.8: Google map of k-means clusters for the city - Pittsburgh, USA

benchmark a reviewer has to meet to be considered an Elite, and each member has to be re-approved by the Council each year. We propose using *Knowledge* of a yelper to mark if a yelper is actually someone, aware about the category related to a business.

A user is considered to have *Knowledge* for a business if: he/she has given a review for the business and has at least 1 friend who reviewed any business which has the same category as the business reviewed by the user. The friends of the user might be living in another location, or reviewed other businesses with the same category, that doesn't affect anything.

First, we get a list of users who have given a review for a business. We then get the list of friends for each user. For each friend of a user we find whether there is at least one review by a friend in any business having exactly the same category as that of the business reviewed by the user. If we are able to find 2 such friends of the user then that user is having *Knowledge* about the business.

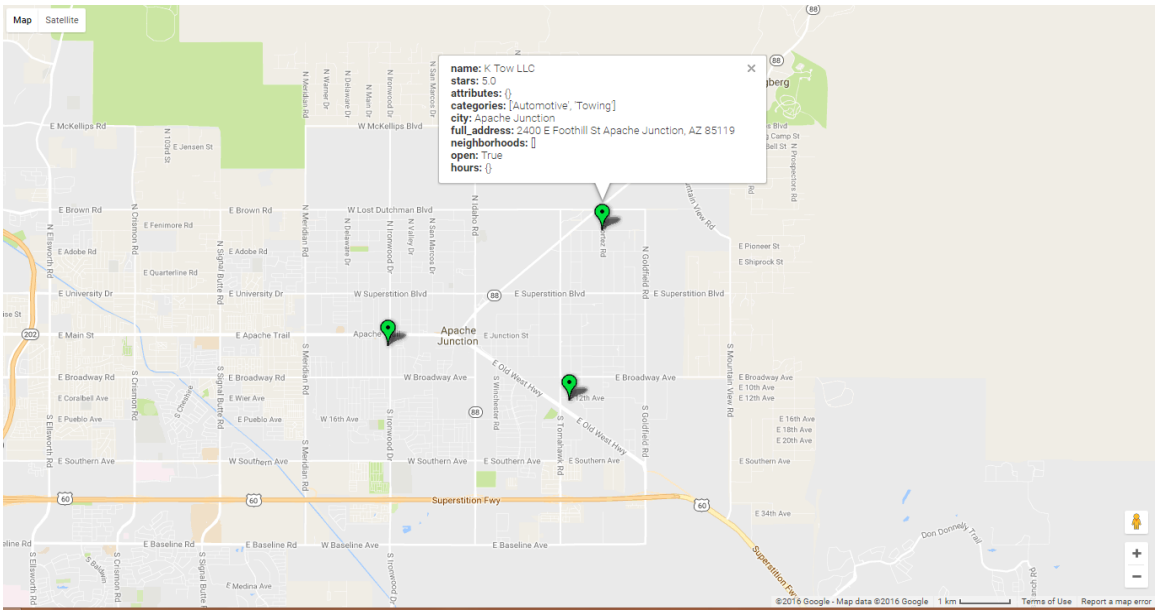

Figure 3.9: Feature map of Car Towing businesses in the City - Apache Junction

Figure 3.13 shows the reviewer's who will be marked as *'Knowledgeable users'* about a local sushi restaurant. For example if we are interested in knowing about a local sushi restaurant with category ('Sushi', 'Restaurant'): As is clearly visible in the above figure, A is friends with D and E who have also given at least one review in the same category. Thus A's review about the local sushi restaurant will be selected. User B may also have reviewed heavily on the same category; however that does not make him an expert. Not unless he is connected to *at least 2 friends* who have given their reviews of a business having exactly the same category. User C gave only one review and has no friends. In the category ('Sushi', 'Restaurant') there is another user F who has given reviews who may or may not have friends in the same category, which is not important. As there is no link which can be established from Business X to F, this user is not selected. Thus only A's review will be selected. Out of 77,445 businesses we found 13 businesses with knowledgeable users. In total we found 13 users with knowledge.

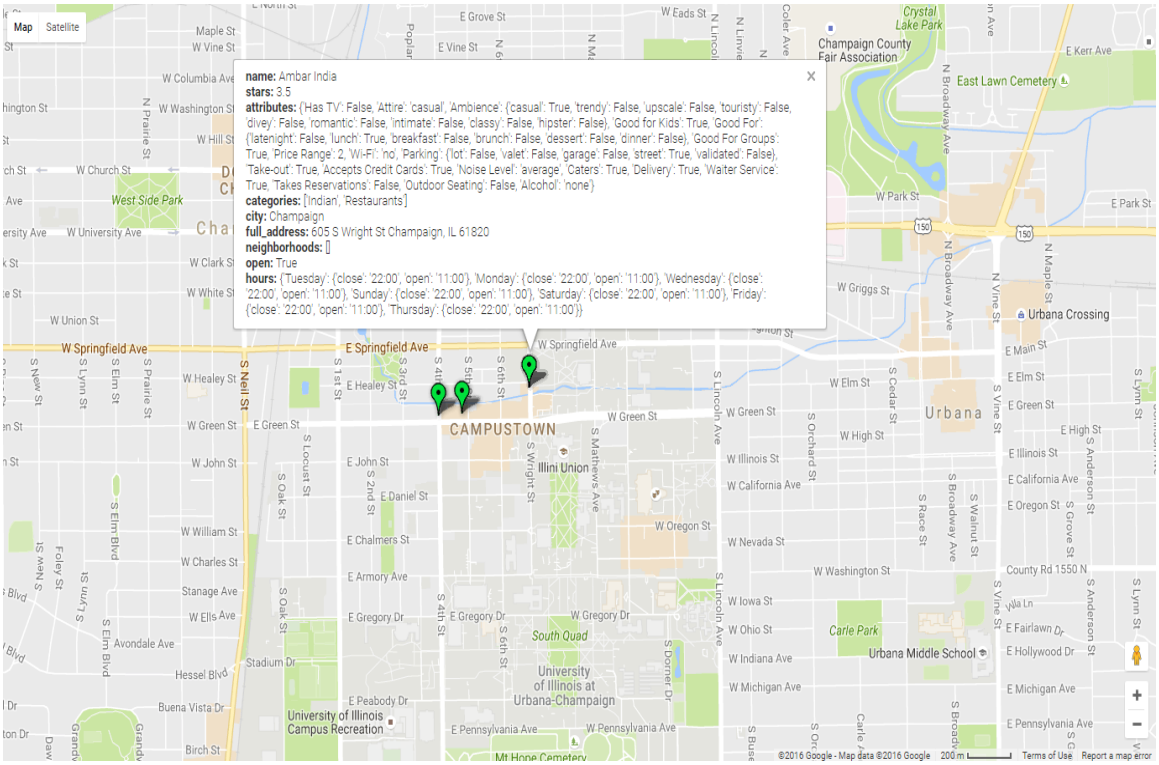

Figure 3.10: Feature map of Indian restaurants in the City - Champaign

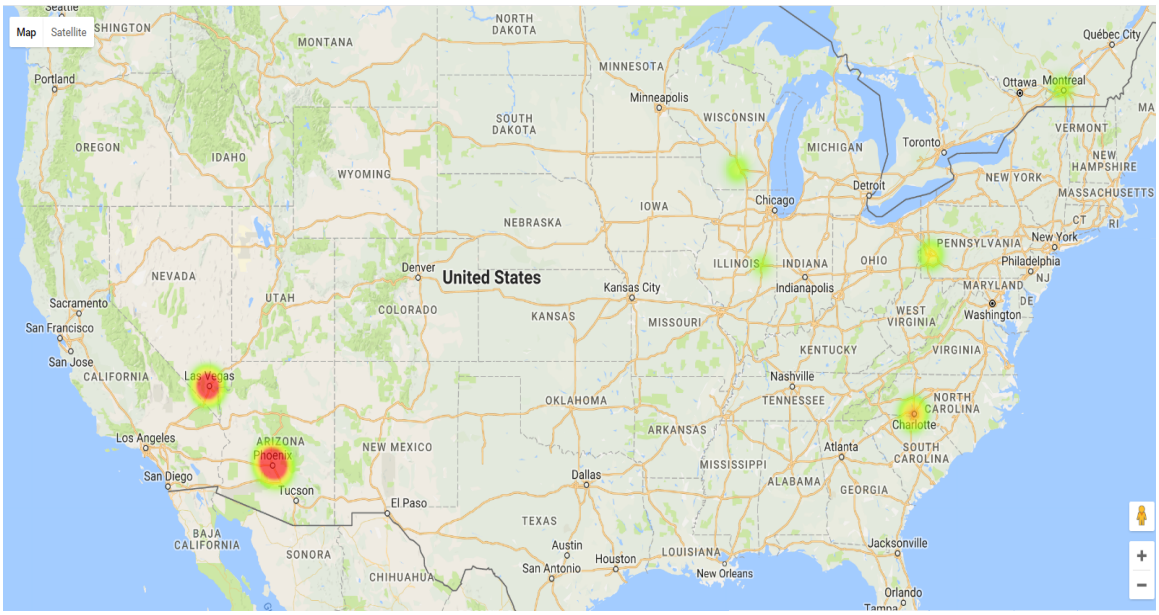

Figure 3.11: Heat-map of 1-star rated businesses

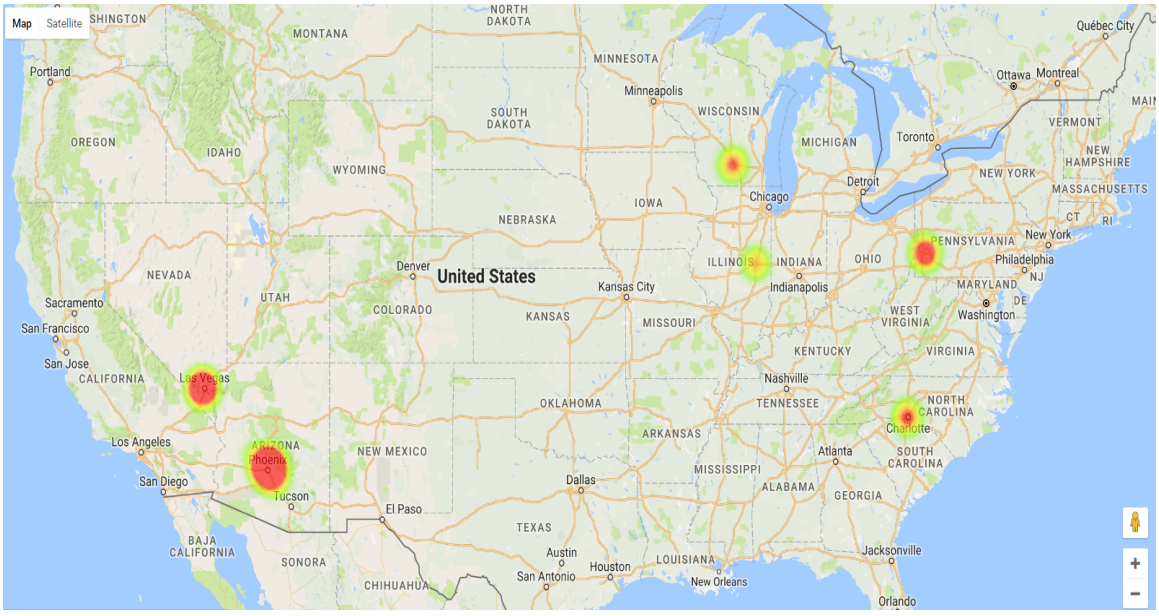

Figure 3.12: Heat-map of 5-star rated businesses

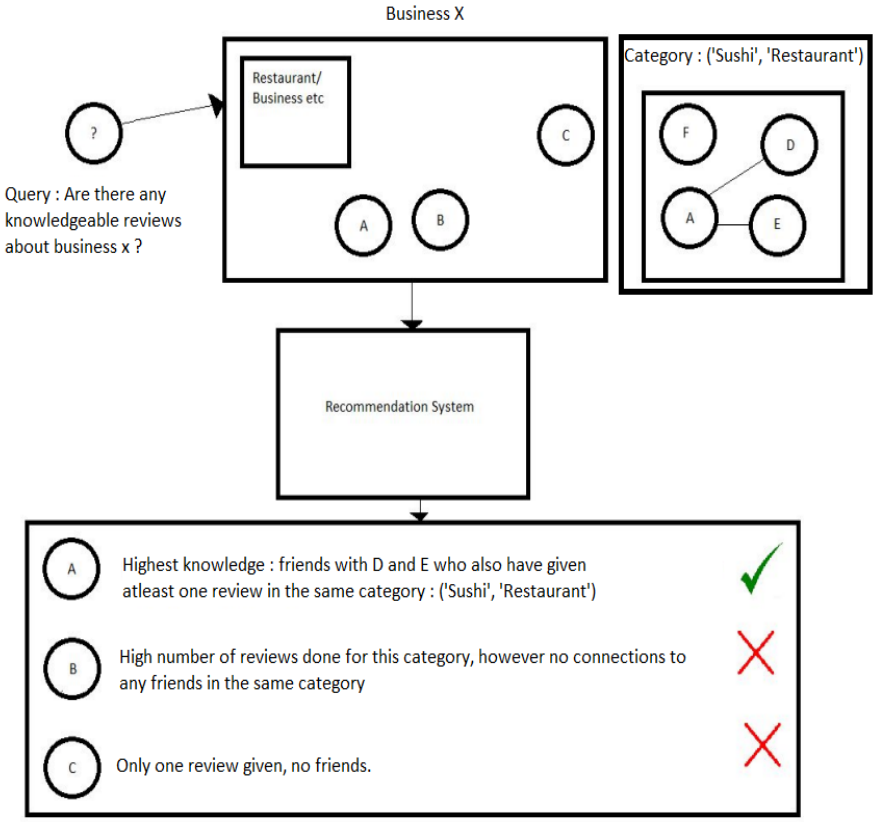

Figure 3.13: Recommendation by Knowledge of a Yelper

Chapter 4

Natural Language Processing for Prediction of Star–ratings from Reviews

There are 77 million desktop and 72 million mobile average monthly unique visitors as measured by Google Analytics, on an average monthly basis over a given three-month period. There are 115 million cumulative reviews contributed since inception. Thus doing sentiment analysis of these reviews using Python's NLTK package[7] is very significant. Sentiment analysis, in simple terms, is predicting the sentiment (positive, negative, happy, sad etc.) of a review written by a user.

4.1 Introduction

We first filtered restaurant businesses from all businesses by searching for the string : 'restaurants' inside the category of a business. Recall that a category is a collection of strings which define what the business does.

Next, we used the NLTK[7] to tokenize each review and do sentiment analysis[6]. We did a review level sentiment analysis. We divided the data into two sets : one, training set comprised of 5000 random restaurant reviews and, testing set comprising of 1,343,627 restaurant reviews. The star ratings of the reviews are used as the target labels for the purpose of developing machine learning models for predicting ratings based on sentiment analysis of the reviews.

We implemented a variety of algorithms to compute features of reviews. We implemented two types of word-sentiment association lexicon. We took 5000 random restaurant reviews and generated different classification and regression models based upon the star rating.

Table 4.1: Datasets

Dataset	1-Star	2-Star	3-Star	4-Star	5-Star	Total
Training	9.38%	9.58%	14.9%	29.54%	36.6%	5000
Test	9.72%	9.59%	14.37%	29.92%	36.41%	1,343,627

Using these models we were able to predict the star rating of each review in the test data and use the mean squared error to measure the efficacy of the regression model.

4.2 Sentiment Lexicons

These are a list of terms with an association to positive or negative sentiment. There are broadly two main types of lexicons used : Already existing (or Manually created lexicons) and Automatically created lexicons.

4.2.1 Already existing lexicons

We used the NRC Emotion Lexicon [3] (about 14,000 words) and the Bing Liu Lexicon[1] (about 6,800 words).

4.2.2 Automatically created lexicons

A term can be described as w where w can be unigram, bigram & non–contiguous pairs (unigram–unigram, bigram–bigram, unigram–bigram). The association score for each term w was calculated from these pseudo-labeled reviews as shown below:

Christopher Pott's script was used to normalize the raw reviews after identifying and marking which tokens are emoticons [5]. We added a substring, 'EMO_' in front of every emoticon to be used to identify whether the emoticon is a positive, negative or neutral emoticon. After each review had been normalized using C. Pott's script we then generated the unigrams, bigrams and the non–contiguous pairs from them. For the non–contiguous

pairs we assumed all tokens within a sliding window of size 5 to be contiguous. Non-contiguous pairs were generated within the sliding window. We generated a pseudo—label from the star rating of each review and assigned a label of 'positive' for the star range 4–5, 'neutral' for star 3 and 'negative' for the star range 1–2.

Using the pseudo–label we divided the training data into sets of positive, neutral & negative reviews. For each review we found the terms (unigrams, bigrams & non–contiguous pairs) belonging to positive, neutral & negative class and then merged the sets together. After this we calculated the score for each term. The score was calculated as mentioned below :

We computed a term frequency matrix. This matrix hold all the generated terms from the training data as (as columns) features and all the reviews (as rows) indices. We for example produced 682,123 features for unigram–pairs from our training data. A term frequency matrix holds the frequency of each term (or feature) in each review as the values of the matrix. Next we constructed a Inverse document matrix(IDF) : M. This IDF matrix (M) is a mathematical model which is intended to reflect how important a term is in a review with respect to the collection of terms across all reviews in the training set. We take the entire set of features from the TF-IDF matrix into a list and we break it into single tokens (or unigrams) and for each unigram we check if it's positive (+1) or negative (-1) in Bing and Liu's lexicon[1] sentiment list. We use a regular expression to determine if a token is an emoticon. We parse each token and search for the substring 'EMO_' if it contains it, then we declare it to be an emoticon and score it positive(+1) or negative(-1) using the Afinn[2] python package. We then sum up the scores of tokens in a term to find out if a term has more positive tokens or negative tokens whichever is higher we assign a sentiment of that class to the term. If none of the terms are present in Bing and Liu's lexicon[1] sentiment list and if there are no positive or negative emoticons in a term then we assign a neutral sentiment score of 0, otherwise we assign +1 for positive and -1 for negative sentiment for a term.

We mask out negative scores (-1) from the list sentiment_vector producing a positively skewed matrix, A and similarly we also constructed a negatively skewed matrix, B. Next

we computed dot product of (M,A) and (M,B). We took the transpose of M and converted all entries more than 0 (terms which had a tf-idf score more than 0) to 1. We took the dot product of this matrix with the results (M,A) and (M,B). This gives a positive (positive_score is a positive value) and negative (negative_score is a negative value) skewed score for each term in the list of features. We merged this positive and negative score generated for each term into a combined score by adding the positive and negative scores.

The combined score may be a negative or positive value depending on the negative_score & positive_score of each term. Thus the magnitude of negative_score tells us about the degree of association of a term with a negative class and similarly we can say the same thing about the positive_score for a term.

Entries were generated for unigram pairs, unigram–bigram pairs, and bigram pairs that were not necessarily immediately contiguous in a review. We assumed a window of size 5 tokens to generate non-contiguous pair for all tokens inside the window size, slide the window through the corpus to get all pairs. High-frequency words like 'the', 'to', 'also' etc. were filtered out before processing, these are called stop—words and they have little lexical content which may be useful for classification of a review. The automatic lexicon has entries for 18,754 unigrams, 183,985 bigrams & 3,234,005 non-contiguous pairs.

4.3 Classification/Regression Models: Features

We transformed each review in the training and test data by representing it as a collection of features as mentioned below (each review is considered as a point in space). All reviews are assigned a star—rating of 1-5 by a user for a restaurant. We are predicting the same star rating but by using the lexical features of each review. We represented each review as a feature vector made up of the following groups of features:

- word ngrams: presence or absence of contiguous sequences of 1, 2, 3, and 4 tokens; non-contiguous ngrams (ngrams with one token replaced by *);

- character ngrams: presence or absence of contiguous sequences of 3, 4, and 5 characters;
- all-caps: the number of words with all characters in upper case;
- lexicons: These features were generated for manually created lexicon : NRC Emotion Lexicon and the automatically generated lexicons. Separate feature sets were produced for unigrams, bigrams, and non-contiguous pairs. The lexicon features were created for all tokens in the review. For each token w and emotion or polarity p, we used the sentiment/emotion score; score(w, p) to determine:
  - total count of tokens in the review with $score(w, p) > 0$;
  - $totalscore = \sum_{w \in review} score(w, p)$;
  - $maximalscore = max_{w \in review} score(w, p)$;
  - the score of the last token in the review with $score(w, p) > 0$
- punctuation:
  - the number of contiguous sequences of exclamation marks, question marks, and both exclamation and question marks;
  - whether the last token contains an exclamation or question mark;
- emoticons: The polarity of an emoticon was determined with a regular expression adopted from Christopher Potts tokenizing script[5]. After this, we prepend substring 'EMO_' to every emoticon in a review. This was used to identify which tokens were an emoticon in a review. The emoticons were scored using the Afinn Python package[2]. Positive score indicates a positive class emoticon and negative score indicates a negative class emoticon.
  - presence or absence of positive and negative emoticons at any position in the review;
  - whether the last token is a positive or negative emoticon;

- elongated words: the number of words with one character repeated more than two times, for example, badddddddddddddddddd;

- negation: : The number of negated contexts. A negated context is a segment of a review that starts with a negation word (e.g., no, shouldnt) and ends with one of the punctuation marks: ,, ., :, ;, !, ?. The regular expression to identify negation words was adopted from Christopher Potts sentiment tutorial[4]. Each review was broken into sentences and then within each sentence the presence of a negation was detected. The total count of the presence of negated contexts across all sentences for a review was calculated and stored as this feature.

4.4 Result of prediction

We trained Linear regression, SGD, ElasticNet & Ridge regression models on the set of 5000 random restaurant reviews. We applied the model to the test set of 1,343,627 unseen restaurant reviews, to predict star ratings. We used the Mean Squared Error (MSE) for the prediction of star ratings from the review for measuring the performance of the models. The results obtained on the test set are shown in Table 5.2.

Table 4.2: Mean Square Error for regression models

Learning Model	MSE	Variance score
Linear Regression	1.67	0.02
SGD	2.86	0.35
ElasticNet	1.69	0.01
Ridge Regression	1.68	0.01

Chapter 5

Conclusions

We performed several kinds of exploratory data analysis on the Yelp data prior to focussing on the task of prediction of star ratings for restaurant reviews. For instance, we performed k-means clustering on the location attributes of the businesses in the city of Pittsburgh and computed the nature of each cluster by subdividing each category of a business into subcategories and taking the top 5 subcategories within each cluster. Thus each cluster was further divided into many pieces depending on the nature of the cluster. We represented this data on a scatter map and google maps. We also computed the knowledge of each yelper and used that to characterize trend setters.

The bulk of the thesis was concerned with developing machine learning models for analyzing sentiments in restaurant reviews. To this end, we built Linear, SGD, ElasticNet, Ridge regression models for sentiment analysis for the reviews using the star ratings as our gold standard. We computed a variety of features based upon the sentiment lexicon. Using the models listed above, we were able to predict the star rating of each review in the test data and used mean squared error to measure each model. Future work would aim to improve the models substantially by considering auxiliary data and latent semantic analysis, for instance, by integrating the photographs provided in the dataset along with reviews for sentiment analysis. Another interesting direction involves a more sophisticated analysis of trend setters (these are consequential users whose reviews roughly track positive or negative trends), and their influence on the best & worst businesses (based upon the average growth rate for a business calculated across multiple years and the number of reviews done by each user for that business' categories). These trend setters could be used to determine which users are affecting a business most significantly over time.

References

[1] Opinion lexicon (or sentiment lexicon). https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html.

[2] Aba-Sah Dadzie Mariann Hardey Matthew Rowe, Milan Stankovic. Finn rup nielsen, a new ANEW: evaluation of a word list for sentiment analysis in microblogs , Proceedings of the ESWC2011 Workshop on making sense of microposts: Big things come in small packages 718 in CEUR Workshop Proceedings: 93-98. may 2011. http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/6006/pdf/imm6006.pdf.

[3] Saif M. Mohammad and Peter D. Turney. Crowdsourcing a word-emotion association lexicon. 29(3):436–465, 2013.

[4] Christopher Potts. Sentiment symposium tutorial: Linguistic structure. http://sentiment.christopherpotts.net/lingstruc.html, annotate = This article was referenced for identifying negated contexts in reviews.

[5] Christopher Potts. Sentiment symposium tutorial: Tokenizing. http://sentiment.christopherpotts.net/tokenizing.html.

[6] Svetlana Kiritchenko Saif M. Mohammad and Xiaodan Zhu. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. http://www.saifmohammad.com/WebDocs/sentimentMKZ.pdf.

[7] Steven Bird, Ewan Klein, and Edward Loper. Natural language processing with python: Analyzing text with the natural language toolkit. http://www.nltk.org/book/.