Table of Contents

Chapter 1

Introduction

Sentiment analysis, the study of complex human affections and subjective information, is a special focus in text mining and natural language processing. With the widespread use of social media in the digital age, sentiment analysis has become an increasingly useful tool for opinion mining and monitoring in different field. Among its many applications, the extraction of sentiments and opinions in user-generated reviews such as reviews of products, movies, or restaurants has engaged much interest given its representation of a direct "voice" of customers and the value embedded in and beneath the words themselves.

While analyses of these voice-of-the-customer (VOC) materials mainly focus on the classification of sentiment polarity at the document level, reviews rarely embodies a single sentiment towards a particular object or entity, but often times involve complex, multi-level, and sometimes contradicting sentiments towards multiple aspects of the same entity. For example, a restaurant review may have an overall positive sentiment towards the experience as a whole, but more specifically a particularly positive sentiment towards the service, neutral towards ambience, and even negative towards the food. These common aspects and their associated sentiments are key to understanding the review and the reviewed entity and can be of great use in many application cases including personalization.

This report proposes a potential procedure for extracting such common topics using topic modeling, illustrated with the Yelp dataset of restaurant reviews. Next steps regarding the aggregation of sentiments associated with these topics will also be discussed.

Chapter 2

Methodology

2.1 Data Preprocessing

Standard text data cleaning procedure was followed for the preprocessing of the review corpus: all characters were transformed to lowercase, punctuations and numbers were removed, stop words such as "I", "me", "she", "is" (see Appendix: stopwords for the full list) were also removed, whitespaces were stripped and trimmed, and all words were stemmed using Porter stemming algorithm (implemented in R through the stemDocument function in the tm package).

2.2 N-Gramming

One optional yet particularly helpful preprocessing step was n-gramming. It captures word sequences that are better perceived or carry more meaningful information as a whole. For example, "White House" should be perceived as a single token instead of separately as "white" and "house", and similarly "President of the United States" makes more sense as a whole. N-gramming is especially helpful when identifying word sequences that are specific to the context of the corpus. In the previous example, it would be crucial to identify "White House" and "President of the United States" as n-grams in a corpus consists of political blogs. In our case where we have a corpus of restaurant reviews, we can expect to see context-specific n-grams such as "Mac 'n Cheese", "highly recommend", "great service", and even "can't wait to go back".

We adopted the probabilistic approach for identifying these n-grams where the conditional probability of seeing the ith word given the (i-1)th word. The n-gramming implementation code was modified based on the NGramming Processing Script (c) by Teague Henry. The procedure in this code is for each consecutive bigram sequence (word1, word2):

1. Calculate count1 = number of appearances of word1 in the corpus;

2. Calculate prop2 = appearance rate of word2 in all non-word1 words in the corpus;

3. Calculate count_bi = number of appearance of bigram word1_word2 in the corpus;

4. Compute p_value = Pr(N count_bi) where N is the total number of consecutive co-occurrences of word1_word2, N ~ Binomial(count1, prop2);

5. If p-value < 0.01, then we reject the null hypothesis that the co-occurrences happened by random and identify word1_word2 as a meaningful bigram.

The above procedure is repeated again after first run to identify trigrams and larger n-grams.

In terms of the cutoff threshold for identifying meaningful n-grams, both count and p-values were considered. While p-value cutoff has comparatively more consistent performance, it alone would include bigrams with neglectable occurrences (eg. appeared only 2 times in the entire corpus) and thus contribute minimal information. As a result, a hybrid cutoff using both a p-value cutoff of 0.01 and empirically-set count cutoffs of 100 for bigrams and 40 for trigrams was adopted for our corpus.

All identified n-grams will be replaced by an integrated token of the original words in the corpus, where bigrams are connected with "_" in between and trigrams with ".". For example, after all pre-processing steps and n-gramming, "White House" would become "white_hous", and "Mac 'n Cheese" would be "mac_n.chees."

2.3 Topic Extraction

2.3.1 Latent Dirichlet Allocation (LDA)

To identify common topics in our corpus, we will first experiment with the topic modeling approach, or more specifically Latent Dirichlet Allocation (LDA). LDA is a Bayesian generative topic model based on the assumption that each document, as a collection of words, is a mixture of a certain number of topics, and that the occurrence of each word in that document can be attributed to one of its topics. In terms of denotations, the entire corpus is a set of documents $\{D_1, ..., Dm\}$, and the words within a document are denoted as $D_i = \{w_{i1}, ..., w_{in_i}\}$, $w_ij \in W$ where W is a finite vocabulary set of size V. Suppose we assume the entire corpus is a mixture of K topics, the data generative process with LDA is as follows:

1. For each topic $k \in \{1, ..., K\}$,

a. Draw a topic-word proportion vector $\phi_k \sim Dirichlet(\alpha)$

2. For each document D_i,

a. Draw a document-topic proportion vector $\theta_i \sim Dirichlet(\beta)$

b. For each word w_{ij},

i. Draw a topic assignment $z_j \sim Multinomial(\theta_i), z_j \in \{1, ..., K\}$

ii. Draw a word from this topic $w_{ij} \sim Multinomial(\phi_k), w_{ij} \in \{1, ..., V\}$

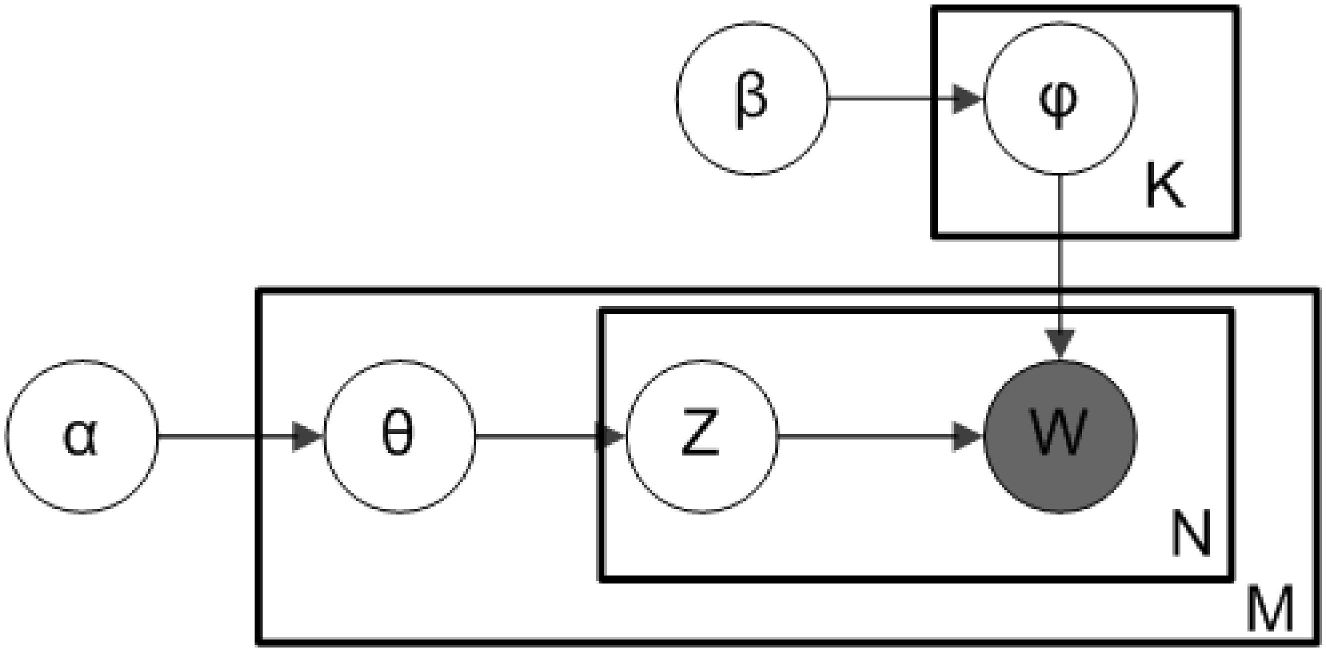

Figure 2.1: LDA data generative process (Wikipedia)

For our corpus, currently the number of topics is manually set at 5, so a future step would be determining the best number of topics for the corpus.

2.4 Sentiment Extraction and Aggregation (Spring Semester)

Chapter 3

Results

The Yelp dataset version as of Round 10 (in terms of the Dataset Challenge) contains 4.7 million reviews of 156K businesses, not limited to restaurants and each with corresponding attributes such as business hours, parking, and ambience. For the purpose of this thesis, we restrict to reviews of restaurants in the US and only examine a sample of this subset, which comes to 57,790 reviews of 1,409 restaurant in the US provided by 47,617 users on Yelp.

3.1 Exploratory Data Analysis

Before diving into the analysis of the corpus, we shall first explore the distribution and characteristics of the reviews and the reviewed restaurants. Note that the insights here does not necessarily reflect the actual distribution or characteristics of the reviewed restaurants or reviews on Yelp since our corpus is a sample of the Yelp-provided dataset which is also a not necessarily random sample of the raw reviews.

As shown below in the graphs, 258 out of the 1409 restaurants reviewed in our corpus are located in the state of Arizona, and the majority of the reviewed restaurants also in Nevada, Ohio, North Carolina, and Pennsylvania, with the rest of the restaurants in Wisconsin and Illinois and only a few in South Carolina, California, or New York. The state distribution may later explain the identification of certain location-specific n-grams such as "las_vega." On average each restaurant has 40 reviews in our corpus, with 50% of them having around 10 reviews and 10% having more than 100 reviews. The review-restaurant distribution may influence the choice and validity of any potential segmentation of the corpus for analysis or later the aggregation of sentiments by restaurants. Most reviews in our corpus are relatively short with an average length of 115 words (median at 81 words), while there surely are more lengthy reviews, with about 3,500 reviews (6% of all reviews) spanning from 300 to 1,000 words.

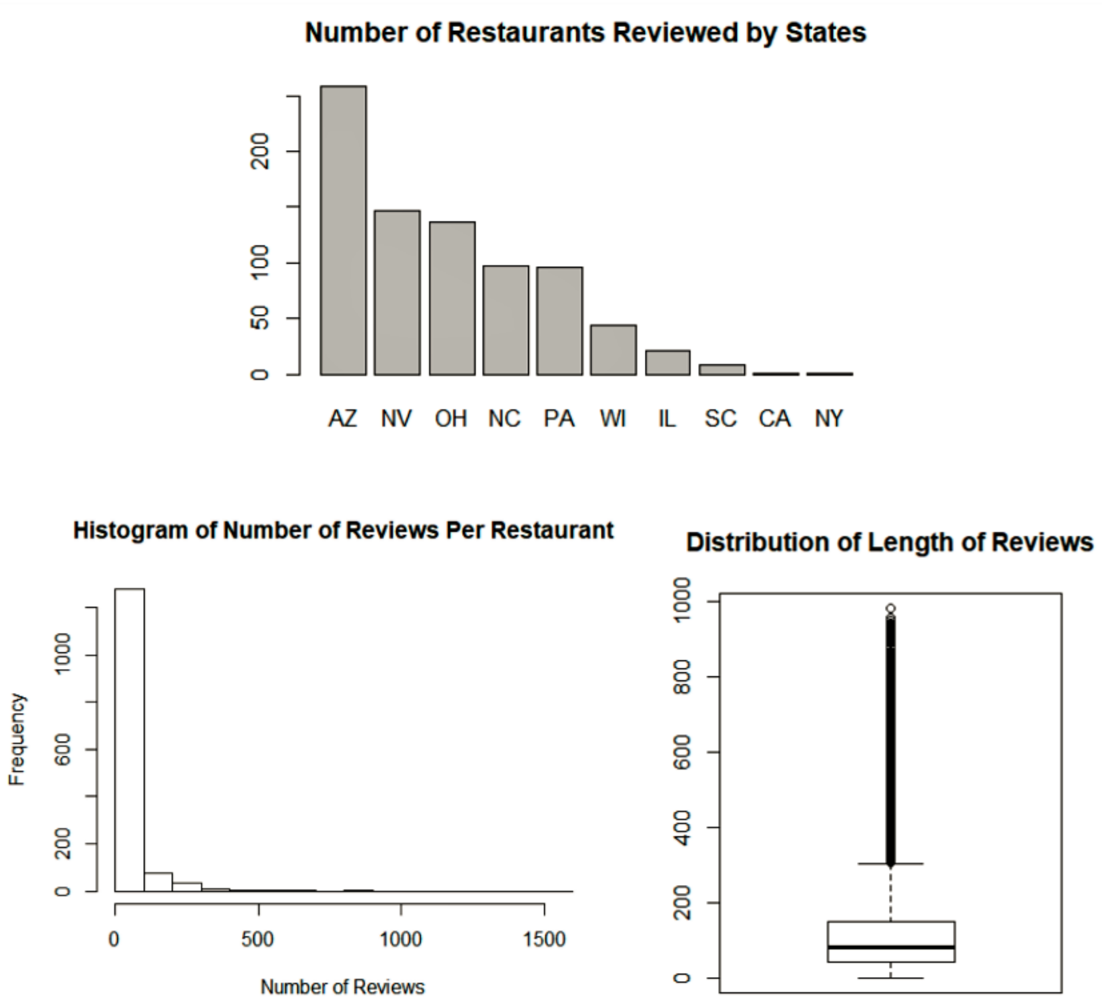

Figure 3.1: Plots illustrating EDA insights on reviews and restaurants

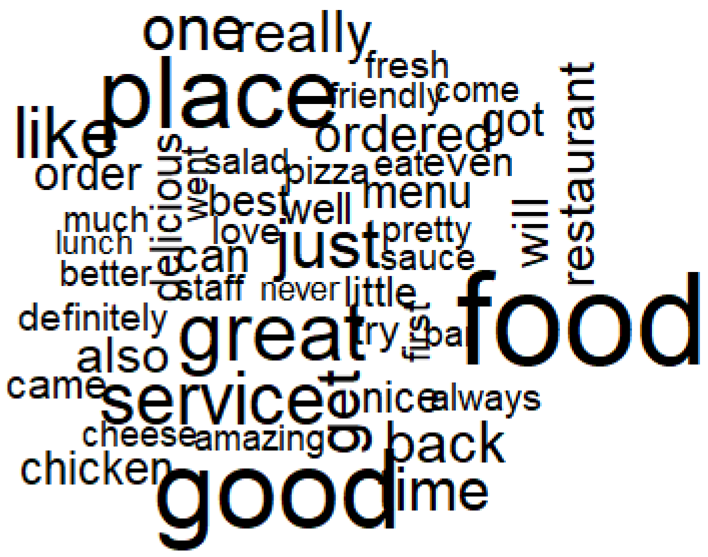

Figure 3.2: Word cloud of 50 most frequent words in corpus (pre-stemming)

3.2 N-Gramming

With the n-gramming model with hybrid cutoff thresholds detailed in the Methodology section, 2148 bigrams and 406 trigrams or larger n-grams were identified. Below is a table of the top and most representative n-grams identified in our corpus of Yelp restaurant reviews.

Table 3.1: Top Context-Specific Bigrams and Trigrams

n-grams	Counts	p-value
go_back	3936	0.000000e+00
high_recommend	1814	0.000000e+00
great_food*	1723	1.060000e-75
happi_hour	1704	1.19e-75
custom_servic	1397	2.73e-26
las_vega	1057	7.60e-201
white_castl**	974	5.79e-152
will_definit.back	482	0.000000e+00
mac_n.chees	222	0.000000e+00
food.pretti_good*	166	1.270000e-121
seat.right_away	113	7.160000e-228
matt_big.breakfast***	101	2.550000e-296
give_place.star	99	2.430000e-184
kung_pao.chicken	54	8.850000e-119
fast_food.chain	40	1.24e-79

Notes on the Bigram Trigram table:
1. (*) Similar bigrams including *"good/great_food/servic/place"* are also common with significant p-values. Same as in the case of trigrams: *"food/servic.pretti/realli/super_good/great"*.
2. (**) Refers to "White Castle", a fast-food chain in Midwestern and Mid-Atlantic regions of the United States. As previously mentioned, the high occurrence of this bigram is likely due to the state distribution of the reviewed restaurants in our corpus.
3. (***) Name of a reviewed restaurant in Phoenix, AZ. Same as above, the high occurrence and significance of p-value of this trigram is also related to our corpus-specific restaurant distributions. But in both cases, identifying the word sequence as one token would carry the right meaningful information and thus beneficial for the analysis.

3.3 Topic Extraction

Regardless of the method used for identifying common topics in the corpus, we may first set our intuitive expectations for the output as a qualitative sanity check. Given that our corpus is a set of reviews on restaurants, commonly reviewed aspects may include food (eg. taste, portion, price), service (eg. timeliness, friendliness), ambience (eg. noise level, suitable occasion), location, etc.. We will see if these targeted aspects can be recovered automatically.

3.3.1 LDA

With number of topics manually set at 5 and Gibbs sampling control slots set at default level (again, a future step could be optimizing these parameters automatically), the topics identified by the LDA model from our corpus along with top words/tokens in each topic is listed below:

LDA topics and associated top terms (entire corpus, stemmed)
Topic 1
Topic 2
Topic 3
Topic 4
Topic 5
place
burger
just
restaur
tabl
locat
chicken
pizza
delici
order
sandwich
sauc
get
meal
ask
day
flavor
eat
salad
us
bar
fri
like
menu
said
alway
order
place
amaz
server

beer
tast
think
dinner
servic
love
good
im
steak
time
friend
meat
price
great
went
great
like
realli
bread
one

The above topics are approximately about restaurant - location and type (Topic 1), food - variety and taste (Topic 2), food - type and price (Topic 3), food - taste and variety (Topic 4), service (Topic 5). While Topic 2 and Topic 4 seems to focus on approximately the same aspects, the former is more about fast-food type of food while the latter seems to be about fine dining.

I have also experimented with running LDA with sub-corpus of each category listed on Yelp (i.e. by cuisine type), and besides showing more category-specific top terms such as "taco", "salsa", and "horchata" in the food topic for the sub-corpus of Mexican restaurants, the segmented corpus did not result in significant improvements in the distinction of topics but rather had worse performance possibly due to less documents in sub-corpus. While segmentation of corpus proves to not be of much help when extracting common topics, it may be helpful later in the step of sentiment aggregation especially we decide to use top terms associated with targeted topics (food, service, ambience, location etc.) as the anchor/center for finding and aggregating sentiment polarity.

Chapter 4

Discussion

Your paper should be an evolving report on the project in all aspects developed so far, in the form of a draft scientific paper. It must be written in RMarkdown or LaTeX with figures included.

Make sure all figures have font sizes and line widths set so that the final pdf versions are properly legible. Presentation (including correctness of mathematical equations, graphics, tables, citations and bibligraphy, as well as prose) should be pristine.

All details of developments of models, code and examples/analyses must be clearly described – sufficient to that a knowledgeable reader will be able to follow the logic and replicate the analysis.

By the end of the first (Fall) semester, you need to develop a readable interim report. In the second (Spring) semester your task is to evolve this paper into a complete write-up of your work, as if intending to consider submitting to a scientific journal.

However primitive the content may seem to be at the start, start writing.

A fairly standard outline is as follows:

- **Title**
- **Abstract**
- **Chapter 1. Introduction** (setting, problem description, citations, etc.)
- **Chapter 2. Literature review**
- **A Next Chapter:** Some papers have one or two chapters, some papers have several. Keep chapters relatively short: Each section should have one focus. For example,
  - **Chapter 3. New Statistical Models** (theory, ideas)
  - **Chapter 4. Some Computational Issues**
  - **Chapter 5. Simulated Data** (evaluation of models)
  - **Chapter 6. Application** (real motivating problem and data)
  - ...
- **Chapter X. Conclusion** (what was done, what was learned, what was good/bad, where research might or could go next)
- **Appendix** (maybe some extra math, details of code)

- **Bibliography** (use bibtex, per the example bib files)

Conclusion

If we don't want Conclusion to have a chapter number next to it, we can add the `{-}` attribute.

More info

And here's some other random info: the first paragraph after a chapter title or section head *shouldn't be* indented, because indents are to tell the reader that you're starting a new paragraph. Since that's obvious after a chapter or section title, proper typesetting doesn't add an indent there.

Appendix A

The First Appendix

This first appendix includes all of the R chunks of code that were hidden throughout the document (using the `include = FALSE` chunk tag) to help with readibility and/or setup.

In the main Rmd file

```
# This chunk ensures that the thesisdowndss package is
# installed and loaded. This thesisdowndss package includes
# the template files for the thesis.
if(!require(devtools))
  install.packages("devtools", repos = "http://cran.rstudio.com")
if(!require(thesisdowndss))
  devtools::install_github("mine-cetinkaya-rundel/thesisdowndss")
library(thesisdowndss)
```

In Chapter ??:

```
# This chunk ensures that the thesisdowndss package is
# installed and loaded. This thesisdowndss package includes
# the template files for the thesis and also two functions
# used for labeling and referencing
if(!require(devtools))
  install.packages("devtools", repos = "http://cran.rstudio.com")
if(!require(dplyr))
    install.packages("dplyr", repos = "http://cran.rstudio.com")
if(!require(ggplot2))
    install.packages("ggplot2", repos = "http://cran.rstudio.com")
if(!require(ggplot2))
    install.packages("bookdown", repos = "http://cran.rstudio.com")
if(!require(thesisdowndss)){
  library(devtools)
  devtools::install_github("mine-cetinkaya-rundel/thesisdowndss")
  }
```

```r
library(thesisdowndss)
# flights <- read.csv("data/flights.csv")
library(topicmodels)
lda <- readRDS("data/LDA_all_ngrammed.rds")
```

Appendix B

The Second Appendix, for Fun

References

Angel, Edward. 2000. *Interactive Computer Graphics : A Top-down Approach with Opengl.* Boston, MA: Addison Wesley Longman.

———. 2001a. *Batch-File Computer Graphics : A Bottom-up Approach with Quicktime.* Boston, MA: Wesley Addison Longman.

———. 2001b. *Test Second Book by Angel.* Boston, MA: Wesley Addison Longman.