Attn: Dr. Sun Aixin

CE/CZ4045 Natural Language Processing

We hereby declare that the attached group assignment has been researched, undertaken, completed and submitted as a collective effort by the group members listed below. We have honored the principles of academic integrity and have upheld Student Code of Academic Conduct in the completion of this work. We understand that if plagiarism is found in the assignment, then lower marks or no marks will be awarded for the assessed work.

Important note: Name must **EXACTLY MATCH** the one printed on your Matriculation Card. Any mismatch leads to **THREE (3)** marks deduction.

Name	Contribution	Signature	Date
CLARITA CHUA WYN KAY	20% (Part 3.2 Most frequent ⟨ Noun - Adjective ⟩ pairs for each rating Code + Report)	[illegible]	23/10/2021
DATTA ANUSHA	20% (Part 3.3 Extraction of Indicative Adjective Phrases Code + Report)	Anusha	23/10/2021
KOTHARI KHUSH MILAN	20% (Part 3.2 Tokenization and Stemming, POS Tagging, Part 3.4 Application Code + Report)	[illegible]	23/10/2021
RAVISHANKAR AMRITA	20% (Part 3.3 Extraction of Indicative Adjective Phrases Code + Report)	Amrita	23/10/2021
SANNABHADTI SHIPRA DEEPAK	20% (Part 3.2 Writing Style Code + Report)	[illegible]	23/10/2021

This assignment was an extremely collaborative effort, wherein all members of the team were highly involved and consistently contributing to all other sections with much overlap in efforts.

Restaurant Review Data Analysis and Processing

CZ4045 Natural Language Processing Assignment (G29), AY21/22 Semester 1

Datta Anusha
U1822948G
anusha007@e.ntu.edu.sg

Ravishankar Amrita
U1822377F
amri0006@e.ntu.edu.sg

Kothari Khush Milan
U1922279J
khush001@e.ntu.edu.sg

Sannabhadti Shipra Deepak
U1822459L
sann0001@e.ntu.edu.sg

Clarita Chua Wyn Kay
U182053J
cchua031@e.ntu.edu.sg

ABSTRACT

This paper observes the uses and challenges of several Natural Language Processing tasks in an end-to-end NLP application. Tokenization and extraction of noun-adjective pairs and indicative adjective phrases are tasks performed on a dataset of customer reviews in Yelp to generate customer insights of these businesses. Additionally, this paper also covers an analysis of the writing style between news articles from ChannelNewsAsia and posts in StackOverflow.

CCS CONCEPTS

• Artificial intelligence → Natural language processing

KEYWORDS

Natural Language Processing, POS Tagging, Stemming, Tokenization, Indicative Adjective Phrases Ranking, Sentiment Analysis

1. INTRODUCTION

Natural Language Processing (NLP) is a discipline under the artificial intelligence field that focuses on analysing and understanding human languages. It has been scaled and utilised in many industries from voice assistant technology to disease prediction. NLP can also help companies gather valuable customer insights from social networks. This paper will be focusing on utilising different NLP tasks to analyse unstructured textual data collected from Yelp, Stackoverflow and ChannelNewsAsia. It will also explore the uses and challenges of using these techniques which includes: (i) tokenization, (ii) POS tagging, (iii) writing style, (iv) extraction of most frequent noun-adjective pairs, (v) extraction of indicative adjective phrases and (vi) sentiment analysis.

1.1 Dataset Description

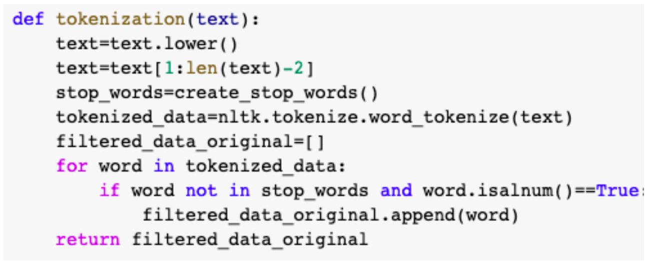
Most of the NLP tasks will be performed on a collection of user reviews posted on Yelp. This dataset contains 15,300 reviews with additional information on review ID, user ID, business ID, stars, date and votes of the review. Votes of the review whether it was funny, useful or cool was omitted from our data analysis.

2. DATA ANALYSIS AND PRE-PROCESSING

2.1 Tokenization and Stemming

2.1.1 Tokenization

The nltk package in python is used to perform tokenization. Before creating tokens, we first check and remove all the necessary stop words. We use the set of stop words to filter out the tokens that occur as stop words in the reviews of the business that was selected. The tokenization function also makes sure that the words are either numbers or characters to filter out all the punctuation marks present in the reviews.



```
def tokenization(text):
    text=text.lower()
    text=text[1:len(text)-2]
    stop_words=create_stop_words()
    tokenized_data=nltk.tokenize.word_tokenize(text)
    filtered_data_original=[]
    for word in tokenized_data:
        if word not in stop_words and word.isalnum()==True:
            filtered_data_original.append(word)
    return filtered_data_original
```

Figure 2.1 Performs tokenization on the text input given

After we perform tokenization, we get a list of tokens without any punctuations in it.

```
['stayed',
 'hotel',
 'saturday',
 'night',
 'cavs',
 'game',
 'drove',
 'past',
 'thought',
 'looked',
 'nice',
 'called',
 'booked',
 'room',
```

Figure 2.2 Result obtained from tokenization

2.1.2 Stemming

To perform stemming on the reviews, we use the PorterStemmer() of the NLTK package. We perform stemming on the tokens we obtained after performing tokenization.

```
['stay',
 'hotel',
 'saturday',
 'night',
 'cav',
 'game',
 'drove',
 'past',
 'thought',
 'look',
 'nice',
 'call',
 'book',
 'room',
 'besid',
```

Figure 2.3 Result obtained after stemming

The output of the stemmed data is different compared to non-stemmed data. This is due to repetitive phrases like 'staying', 'stayed' etc. (seen in Figure 2.3) get reduced to their base word 'stay'. This is why the length of the list of stemmed tokens and non-stemmed tokens is different.

2.1.3 Frequency Distribution

We use the FreqDist() function in the nltk package to calculate the frequency of tokens before and after performing stemming on the reviews of the selected business. The output is a dictionary with the tokens as keys and their frequency as the values. On comparing the output before stemming and after stemming, we can see that the count of words like 'stay' has increased as compared to before stemming where the word was 'staying'. This is because the repeated phrases in stemming are reduced to their base word as mentioned earlier.

```
{'stayed': 33,  {'stay': 100,
 'hotel': 160,   'hotel': 178,
 'saturday': 7,  'saturday': 8,
 'night': 29,    'night': 32,
 'cavs': 3,      'cav': 3,
 'game': 10,     'game': 12,
 'drove': 2,     'drove': 2,
 'past': 2,      'past': 2,
 'thought': 10,  'thought': 11,
 'looked': 5,    'look': 23,
 'nice': 43,     'nice': 43,
 'called': 14,   'call': 23,
 'booked': 3,    'book': 4,
 'room': 155,    'room': 193,
 'besides': 1,   'besid': 1,
```

Figure 2.4 Non-Stemmed vs Stemmed Words Frequency

We can see the difference in the frequencies of the top 10 most used tokens as well. The count in case of stemmed tokens is higher as compared to that in non-stemmed tokens. The token that is most used is different as well. In the case of non-stemmed data it is 'hotel' while in the case of stemmed data it is 'room'. This is due to the reduction of repetitive phrases, originating from the base word 'room', by performing stemming.

```
[('hotel', 160), [('room', 193),
 ('room', 155),   ('hotel', 178),
 ('great', 65),   ('stay', 100),
 ('staff', 57),   ('great', 65),
 ('service', 53)  ('staff', 57),
 ('stay', 50),    ('bar', 54),
 ('one', 47),     ('servic', 53),
 ('would', 45),   ('one', 48),
 ('bar', 44),     ('would', 45),
 ('nice', 43)]    ('nice', 43)]
```

Figure 2.5 Top 10 Highest used tokens in Non-stemmed and Stemmed data respectively

Lastly, for better analysis and visualisation, we create a word cloud of both non-stemmed review's frequency distribution and stemmed review's frequency distribution. The words which are used most often can be seen more prominently as compared to those that are not used as frequently.

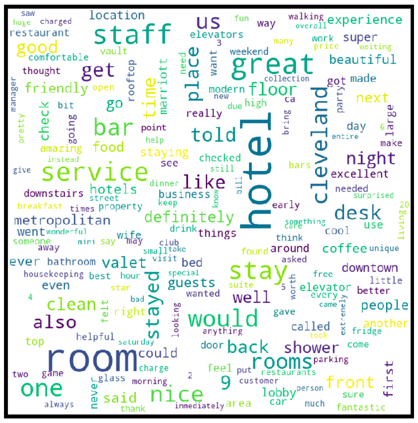


Figure 2.6 Word Cloud of Non-Stemmed Reviews

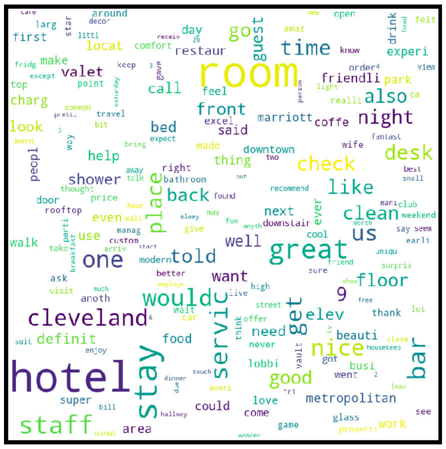


Figure 2.7 Word cloud of Stemmed Reviews

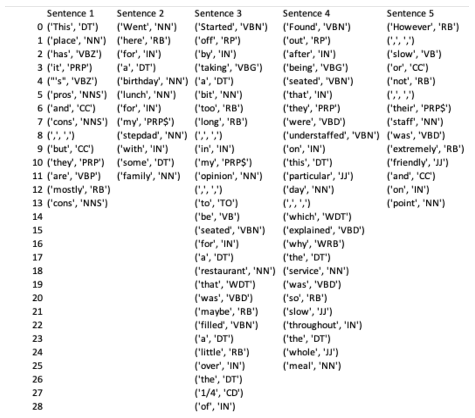
2.2 POS Tagging

For POS tagging, 5 sentences are randomly selected and we use two different packages to compare the POS Tagging output obtained. We have used the nltk and spacy package in python to perform POS tagging. The 5 sentences selected are:

1. *'This place has it's pros and cons, but they are mostly cons'.*
2. *' Went here for a birthday lunch for my stepdad with some family.'.*
3. *' Started off by taking a bit too long, in my opinion, to be seated for a restaurant that was maybe filled a little over the ¼ of it's capacity'.*
4. *' Found out after being seated that they were understaffed on this particular day, which explained why the service was so slow throughout the whole meal.'.*
5. *' However, slow or not, their staff was extremely friendly and on point.'.*

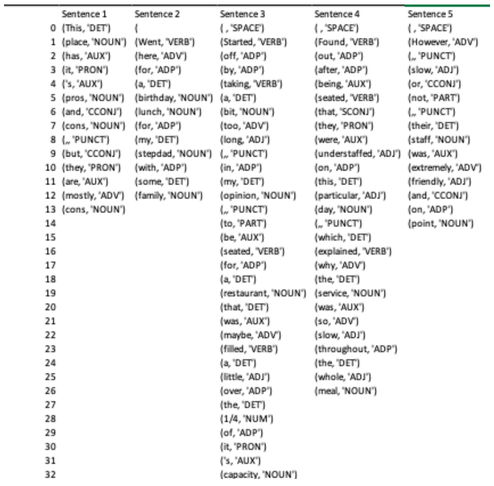
The POS tagger, for both nltk and spacy, largely tags each word correctly. We will compare a sentence tagged by the two packages word by word to see the difference or similarity between them.

For the POS tagging of the first sentence the nltk package tags **" 's "** as a **VBZ** directly whereas the spacy package is more detailed and gives it the **auxiliary verb** tag which is more appropriate in this case. Similarly, nltk is more detailed in specifying whether the noun is plural or singular as seen by the **NNS** tags for **"pros"** and **"cons"** words. However, these same words are given just the **NOUN** tag by the spacy package. Otherwise, both give similar results however, they denote the tags in different formats, for example, the determinant tag is denoted by **DT** in nltk and **DET** in spacy.



	Sentence 1	Sentence 2	Sentence 3	Sentence 4	Sentence 5
0	('This', 'DT')	('Went', 'NN')	('Started', 'VBN')	('Found', 'VBN')	('However', 'RB')
1	('place', 'NN')	('here', 'RB')	('off', 'RP')	('out', 'RP')	(',', ',')
2	('has', 'VBZ')	('for', 'IN')	('by', 'IN')	('after', 'IN')	('slow', 'VB')
3	('it', 'PRP')	('a', 'DT')	('taking', 'VBG')	('being', 'VBG')	('or', 'CC')
4	("'s", 'VBZ')	('birthday', 'NN')	('a', 'DT')	('seated', 'VBN')	('not', 'RB')
5	('pros', 'NNS')	('lunch', 'NN')	('bit', 'NN')	('that', 'IN')	(',', ',')
6	('and', 'CC')	('for', 'IN')	('too', 'RB')	('they', 'PRP')	('their', 'PRP$')
7	('cons', 'NNS')	('my', 'PRP$')	('long', 'RB')	('were', 'VBD')	('staff', 'NN')
8	(',', ',')	('stepdad', 'NN')	(',', ',')	('understaffed', 'VBN')	('was', 'VBD')
9	('but', 'CC')	('with', 'IN')	('in', 'IN')	('on', 'IN')	('extremely', 'RB')
10	('they', 'PRP')	('some', 'DT')	('my', 'PRP$')	('this', 'DT')	('friendly', 'JJ')
11	('are', 'VBP')	('family', 'NN')	('opinion', 'NN')	('particular', 'JJ')	('and', 'CC')
12	('mostly', 'RB')		(',', ',')	('day', 'NN')	('on', 'IN')
13	('cons', 'NNS')		('to', 'TO')	(',', ',')	('point', 'NN')
14			('be', 'VB')	('which', 'WDT')	
15			('seated', 'VBN')	('explained', 'VBD')	
16			('for', 'IN')	('why', 'WRB')	
17			('a', 'DT')	('the', 'DT')	
18			('restaurant', 'NN')	('service', 'NN')	
19			('that', 'WDT')	('was', 'VBD')	
20			('was', 'VBD')	('so', 'RB')	
21			('maybe', 'RB')	('slow', 'JJ')	
22			('filled', 'VBN')	('throughout', 'IN')	
23			('a', 'DT')	('the', 'DT')	
24			('little', 'RB')	('whole', 'JJ')	
25			('over', 'IN')	('meal', 'NN')	
26			('the', 'DT')		
27			('1/4', 'CD')		
28			('of', 'IN')		

Figure 2.8 Results of nltk's POS Tagging



	Sentence 1	Sentence 2	Sentence 3	Sentence 4	Sentence 5
0	(This, 'DET')	(	(, 'SPACE')	(, 'SPACE')	(, 'SPACE')
1	(place, 'NOUN')	(Went, 'VERB')	(Started, 'VERB')	(Found, 'VERB')	(However, 'ADV')
2	(has, 'AUX')	(here, 'ADV')	(off, 'ADP')	(out, 'ADP')	(,, 'PUNCT')
3	(it, 'PRON')	(for, 'ADP')	(by, 'ADP')	(after, 'ADP')	(slow, 'ADJ')
4	('s, 'AUX')	(a, 'DET')	(taking, 'VERB')	(being, 'AUX')	(or, 'CCONJ')
5	(pros, 'NOUN')	(birthday, 'NOUN')	(a, 'DET')	(seated, 'VERB')	(not, 'PART')
6	(and, 'CCONJ')	(lunch, 'NOUN')	(bit, 'NOUN')	(that, 'SCONJ')	(,, 'PUNCT')
7	(cons, 'NOUN')	(for, 'ADP')	(too, 'ADV')	(they, 'PRON')	(their, 'DET')
8	(,, 'PUNCT')	(my, 'DET')	(long, 'ADJ')	(were, 'AUX')	(staff, 'NOUN')
9	(but, 'CCONJ')	(stepdad, 'NOUN')	(,, 'PUNCT')	(understaffed, 'ADJ')	(was, 'AUX')
10	(they, 'PRON')	(with, 'ADP')	(in, 'ADP')	(on, 'ADP')	(extremely, 'ADV')
11	(are, 'AUX')	(some, 'DET')	(my, 'DET')	(this, 'DET')	(friendly, 'ADJ')
12	(mostly, 'ADV')	(family, 'NOUN')	(opinion, 'NOUN')	(particular, 'ADJ')	(and, 'CCONJ')
13	(cons, 'NOUN')		(,, 'PUNCT')	(day, 'NOUN')	(on, 'ADP')
14			(to, 'PART')	(,, 'PUNCT')	(point, 'NOUN')
15			(be, 'AUX')	(which, 'DET')	
16			(seated, 'VERB')	(explained, 'VERB')	
17			(for, 'ADP')	(why, 'ADV')	
18			(a, 'DET')	(the, 'DET')	
19			(restaurant, 'NOUN')	(service, 'NOUN')	
20			(that, 'DET')	(was, 'AUX')	
21			(was, 'AUX')	(so, 'ADV')	
22			(maybe, 'ADV')	(slow, 'ADJ')	
23			(filled, 'VERB')	(throughout, 'ADP')	
24			(a, 'DET')	(the, 'DET')	
25			(little, 'ADJ')	(whole, 'ADJ')	
26			(over, 'ADP')	(meal, 'NOUN')	
27			(the, 'DET')		
28			(1/4, 'NUM')		
29			(of, 'ADP')		
30			(it, 'PRON')		
31			('s, 'AUX')		
32			(capacity, 'NOUN')		

Figure 2.9 Results of spacy's POS Tagging

2.3 Writing Style

2.3.1 Random Sampling, Web Scraping, Tokenization

Random sampling has been employed to extract text for the purpose of analysing the variation in writing style. The samples include: (i) two posts from StackOverflow, (ii) two posts from Hardwarezone, (iii) two news articles from ChannelNewsAsia (CNA). The selected posts/articles are:

1. https://stackoverflow.com/questions/20940979/what-is-an-indexoutofrangeexception-argumentoutofrangeexception-and-how-do-i-f
2. https://stackoverflow.com/questions/952914/how-to-make-a-flat-list-out-of-a-list-of-lists
3. https://www.hardwarezone.com.sg/tech-news-apple-blacklists-epic-games-and-disallow-fortnite-back-app-store
4. https://www.hardwarezone.com.sg/tech-news-m2-powered-macbook-air-rumoured-launch-q3-2022

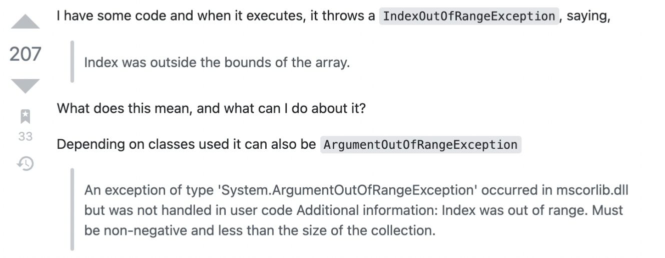
5. https://www.channelnewsasia.com/world/australia-covid-19-sydney-unvaccinated-lockdown-freedom-2206346
6. https://www.channelnewsasia.com/singapore/urban-farms-pesticide-free-vegetables-accreditation-scheme-singapore-2204351

The text elements from these articles and posts are then extracted for analysis. During this process, it was found that the Stack Overflow posts often had code elements in addition to text. These were extracted separately (for the most part), to prevent interference with the text analysis. The extracted text is then tokenized using the respective NLTK library.

2.3.2 Rudimentary Proper Noun Analysis

A quick grammar check is performed manually utilising the tokenized text. This involves checking the first letter of each word for capitalization, ensuring that the previous token is an appropriate separator ([.] , [?] , [!] , [:] , [“]) and that the word in question is not the personal pronoun “I”. The highlighted method uses grammar rules to filter proper nouns.

While the rudimentary proper noun flagging method above correctly flags only contextual proper nouns for the HardwareZone and CNA Articles, it flagged 1 proper noun ('Python') and 5 irregularly capitalised words ('IndexOutOfRangeException', 'Index', 'ArgumentOutOfRangeException', 'An', 'Additional') in Stack Overflow. On further inspection, it is evident that this is due to the format and context of the piece of text.


207

33

I have some code and when it executes, it throws a `IndexOutOfRangeException`, saying,

> Index was outside the bounds of the array.

What does this mean, and what can I do about it?

Depending on classes used it can also be `ArgumentOutOfRangeException`

> An exception of type 'System.ArgumentOutOfRangeException' occurred in mscorlib.dll but was not handled in user code Additional information: Index was out of range. Must be non-negative and less than the size of the collection.

Figure 2.10 The Relevant Stack Overflow Text
(https://stackoverflow.com/questions/20940979/what-is-an-indexoutofrangeexception-argumentoutofrangeexception-and-how-do-i-f)

'IndexOutOfRangeException' and 'ArgumentOutOfRangeException' are not regular words in the English language. They are exceptions returned by the compiler (which makes sense as Stack Overflow is a programming website). End of sentence punctuation ('.' or '!' or '?') is not adequately used before 'Index' and 'An' because the author embeds these sentences as a response on the page. In the case of 'Additional' the author of the text seems to have made a grammatical error (which is not uncommon on Stack Overflow, as it is an informal information source).

2.3.3 Stopwords

Stopwords are filtered using the NLTK corpus. Further analysis is done to capture the frequency and percentage of stopword use in each type of text (Stack Overflow, HardwareZone, CNA). The results of this analysis are plotted in pie charts, for comparison.

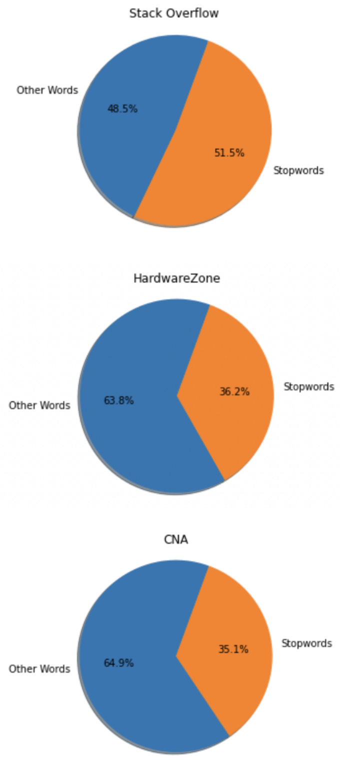


Figure 2.11 Stopword Comparison

We see that the Stack Overflow text contains a greater proportion of Stopwords when compared to that in the HardwareZone and CNA articles. This writing style may indicate that the Stack Overflow sentences are less dense i.e. have less specific information in its body of text.

2.3.4 POS Tagging

POS tagging is performed on the tokenized words. The base step is performed using the NLTK POS tagging library.
Next, the Stack Overflow POS tags are first cleaned using the information received from the rudimentary proper noun analysis. Here, the words that were incorrectly tagged as proper nouns are tagged manually. Additionally, the embedded pieces of code are extracted from the Stack Overflow text to be tagged as ‘CODE’. Here checks are made to flag tokens in camelcase, for loops, if

statements and functions/methods/variables. Following this, the code elements extracted during the initial web scrape are appropriately tagged as 'CODE' and appended to the list of Stack Overflow tagged tokens. Finally, the stopwords are extracted for comparison.

2.3.5 POS Analysis

The POS counts are obtained for the respective pieces of texts (with and without stopwords). These counts are plotted for the purposes of comparison, so as to perform further analysis.

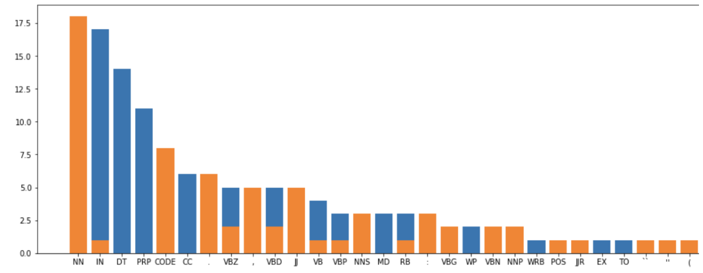


Figure 2.12 Stack Overflow POS Counts

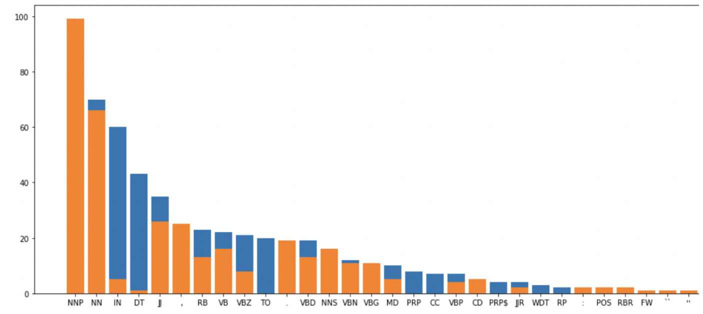


Figure 2.13 HardwareZone POS Counts

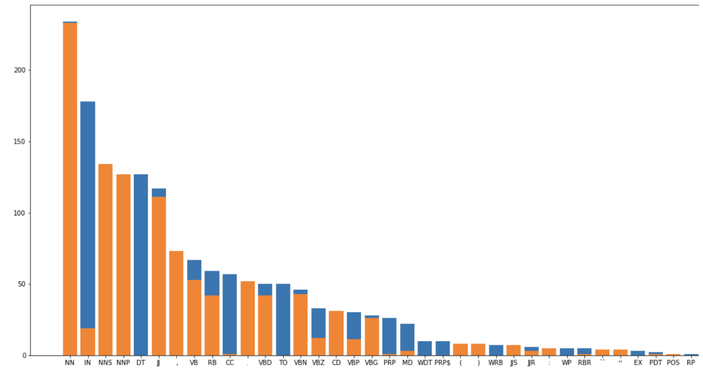


Figure 2.14 CNA POS Counts

Note that the counts inclusive of stopwords are in blue, and the counts exclusive of stopwords are in orange.

On inspection a noticeable disparity was seen in the counts of personal pronouns and proper nouns. For comparison, the personal pronoun and proper noun counts were plotted in pie charts. Personal pronouns are plotted against the tagged elements including stopwords (as many personal pronouns are included in the stopword corpus) while the proper nouns are plotted against the tagged elements excluding stopwords (as stop words are irrelevant in this case).

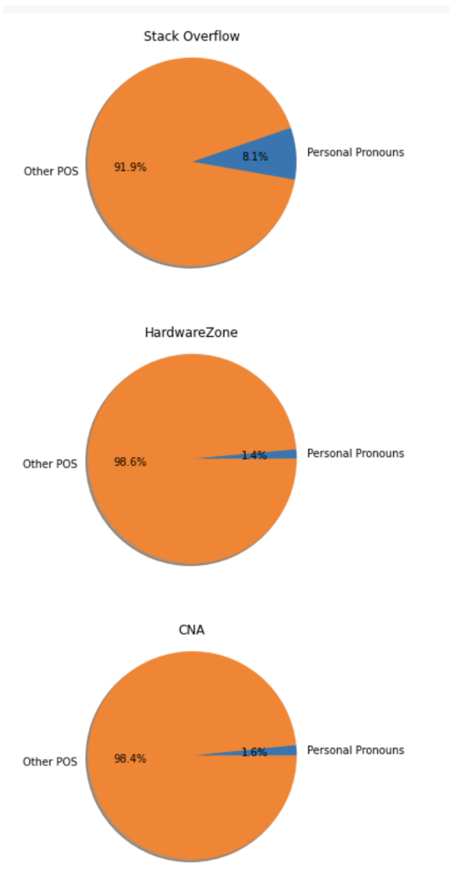


Figure 2.15 Comparative Analysis of Personal Pronoun Usage

We notice that personal pronouns are used with far greater frequency in the Stack Overflow text. There is, therefore, an indication of a more individualistic/personal focussed writing style in the StackOverflow text as opposed to the HardwareZone/CNA articles.

This hypothesis can be confirmed by looking further into the nature of the texts – Stack Overflow is a forum wherein users pose technical questions based on their individual concerns. Therefore, there is naturally a heavy focus on individual experience, explaining the high frequency use of personal pronouns. In contrast, HardwareZone and CNA handle journalistic issues from an outside objective perspective, making there little use for personal pronouns.

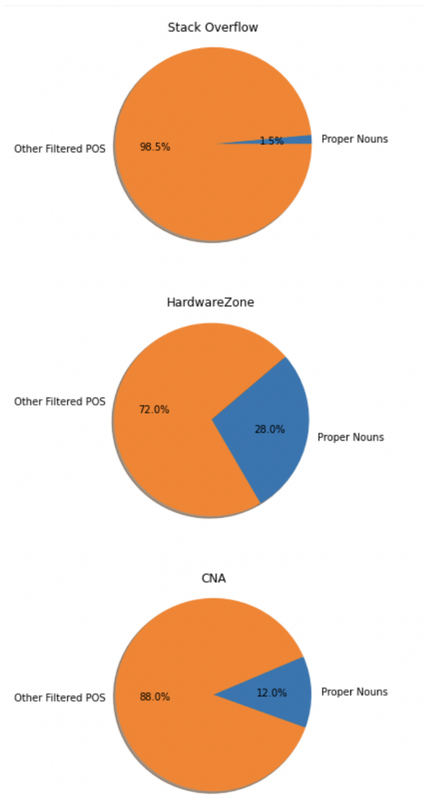


Figure 2.16 Comparative Analysis of Proper Noun Usage

There seems to be a higher frequency of proper nouns in the HardwareZone Articles, followed by the CNA Articles. There are very few proper nouns in the Stack Overflow text.

This can again be attributed to the nature of the texts – The HardwareZone articles reference important company developments and launches in the technological sphere. Here the writing style is very impersonal and information heavy. The high frequency of proper nouns indicates a focus on third-party referential detailed information. The CNA articles also have a focus on third-party, high-volume information, but the proper noun usage in their writing style indicates that certain information may not be as specific (when compared to the HardwareZone article). The Stack Overflow text does not relay or reference nearly as many third parties as the previous two pieces of text, as it is not a diverse (in terms of information) or objective information source.

2.4 Most frequent Noun-Adjective Pairs for each rating

Most reviews are structured with a noun subject and an adjective to describe the subject. By extracting frequent noun adjective pairs from reviews, businesses are able to understand the overall consensus of how their customers view them and what particular menu items or service do customers find unsatisfactory.

2.4.1 Noun Adjective Pairs

To extract a noun adjective pair in a sentence, a noun is first detected and the adjective related to the noun is retrieved and added to the pair. There can be multiple noun adjective pairs in a review and examples of these pairs are service-great and food-bad.

The noun-adjective pair extraction was performed utilising the linguistic feature from the spaCy library. spaCy is designed to parse and tag a raw text and their annotations allow us to get fast and accurate syntactic relations between tokens. Every word has exactly one head and to extract a noun-adjective pair, the head's POS tag must be a noun and the syntactic relations of each child to head can either be an adjectival modifier of a noun (amod) and adjectival complement (acomp). The figure below illustrates the syntactic relations of the sentence placed under each arc:

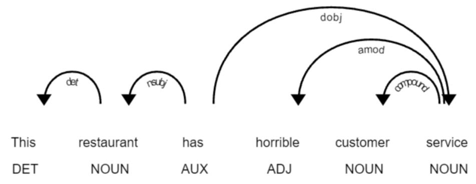
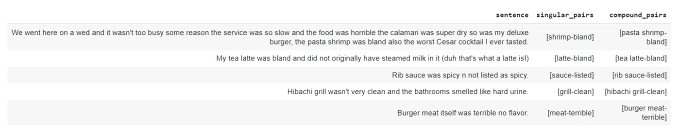
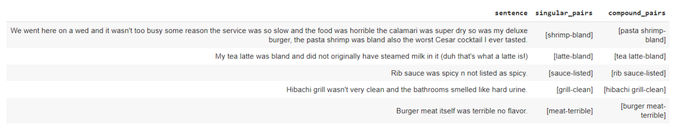


Figure 2.17 Syntactic Relation Tree

From Figure 19, there is one noun-adjective pair extracted and it is service-horrible because service is the only noun with a syntactic relationship with the adjective, horrible in the sentence. There are 2 types of nouns that can be detected, a singular noun and a compound noun. A singular noun is made up of 1 word while a compound noun is a noun made up of 2 or more words such as service and customer service.



sentence	singular_pairs	compound_pairs
We went here on a wed and it wasn't too busy some reason the service was so slow and the food was horrible the calamari was super dry so was my deluxe burger, the pasta shrimp was bland also the worst Cesar cocktail I ever tasted.	[shrimp-bland]	[pasta shrimp-bland]
My tea latte was bland and did not originally have steamed milk in it (duh that's what a latte is!)	[latte-bland]	[tea latte-bland]
Rib sauce was spicy n not listed as spicy.	[sauce-listed]	[rib sauce-listed]
Hibachi grill wasn't very clean and the bathrooms smelled like hard urine.	[grill-clean]	[hibachi grill-clean]
Burger meat itself was terrible no flavor.	[meat-terrible]	[burger meat-terrible]



sentence	singular_pairs	compound_pairs
We went here on a wed and it wasn't too busy some reason the service was so slow and the food was horrible the calamari was super dry so was my deluxe burger, the pasta shrimp was bland also the worst Cesar cocktail I ever tasted.	[shrimp-bland]	[pasta shrimp-bland]
My tea latte was bland and did not originally have steamed milk in it (duh that's what a latte is!)	[latte-bland]	[tea latte-bland]
Rib sauce was spicy n not listed as spicy.	[sauce-listed]	[rib sauce-listed]
Hibachi grill wasn't very clean and the bathrooms smelled like hard urine.	[grill-clean]	[hibachi grill-clean]
Burger meat itself was terrible no flavor.	[meat-terrible]	[burger meat-terrible]

Figure 2.18 Singular and Compound Nouns

We have created two functions dedicated to extracting compound noun-adjective pairs and singular noun-adjective pairs. Compound noun-adjective pairs have more context as a compound noun is more detailed than a singular noun shown in Figure 20. Therefore, only compound noun-adjective pairs are considered when finding frequent pairs if segments have both singular and compound noun-adjective pairs.

2.4.2 Sentence Segmentation

An initial data exploration shows a wide distribution of word length of the reviews thus segmenting sentences are needed to handle long reviews. Referring to Figure 21, most reviews have a word length of 20 to 80 words.

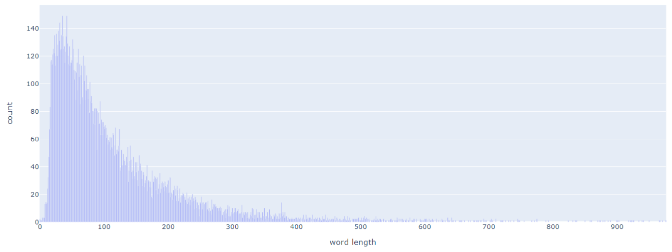
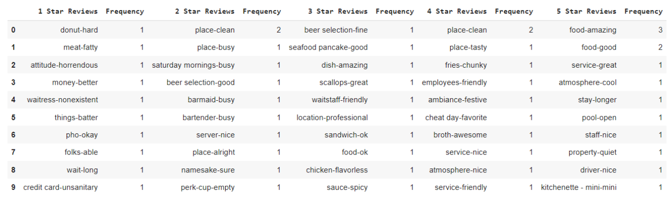


Figure 2.19 Counts of Different Word Length

The spaCy's pipeline component called Sentencizer was used to split reviews into sentences. The Sentencizer implements a simple rule-based strategy to find sentences applying a pre-configuration of punctuations that marks the end of a sentence.

2.4.3 Most Frequent Noun Adjective Pairs for each Rating

To find the most frequent noun adjective pairs of each rating, 50 different business reviews with each rating was first sampled. These reviews are split into segments for simpler analysis and each segment undergoes the noun-adjective pair extraction process. After all noun-adjective pairs are collated, the count of each unique noun-adjective pair is calculated. The top 10 frequent noun-adjective pairs for each batch of reviews grouped by their rating is shown in the Figure 22.



	1 Star Reviews	Frequency	2 Star Reviews	Frequency	3 Star Reviews	Frequency	4 Star Reviews	Frequency	5 Star Reviews	Frequency
0	donut-hard	1	place-clean	2	beer selection-fine	1	place-clean	2	food-amazing	3
1	meat-fatty	1	place-busy	1	seafood pancake-good	1	place-tasty	1	food-good	2
2	attitude-horrendous	1	saturday mornings-busy	1	dish-amazing	1	fries-chunky	1	service-great	1
3	money-better	1	beer selection-good	1	scallops-great	1	employees-friendly	1	atmosphere-cool	1
4	waitress-nonexistent	1	barmaid-busy	1	waitstaff-friendly	1	ambiance-festive	1	stay-longer	1
5	things-batter	1	bartender-busy	1	location-professional	1	cheat day-favorite	1	pool-open	1
6	pho-okay	1	server-nice	1	sandwich-ok	1	broth-awesome	1	staff-nice	1
7	folks-able	1	place-alright	1	food-ok	1	service-nice	1	property-quiet	1
8	wait-long	1	namesake-sure	1	chicken-flavorless	1	atmosphere-nice	1	driver-nice	1
9	credit card-unsanitary	1	perk-cup-empty	1	sauce-spicy	1	service-friendly	1	kitchenette - mini-mini	1

Figure 2.20 Top 10 Frequent Noun-Adjective Pairs of Different Star Reviews

From the top 10 frequent noun-adjective pairs of reviews with 1 star, poor service and poor food quality are common properties for these businesses. The common adjective appearing frequently in reviews with 2 stars is busy which indicates a poor service because the customers are left unattended for a long time. The noun-adjective pairs of reviews with more than 3 stars indicates a strong relationship between customer service and environment. The one factor that distinguishes the higher star ratings is customer service where there are more noun-adjective pairs stating the great service and friendly staff of the businesses. The adjectives of these noun-adjective pairs are also of higher level of praise with words like amazing, great and friendly appearing frequently in reviews with higher than 3 star ratings. Environment is also another factor that affects the review's rating of the business. If businesses provide a cool or festive atmosphere, this will help increase their star ratings as it provides a unique experience for the customers.

2.4.4 Limitations

Although finding frequent noun-adjective pairs can give an idea of how a business is doing, the current method can be improved by adding more context to the noun-adjective pairs such as adding adverbs. Some noun-adjective pairs may not be meaningful when adverbs are not included such as cheese-tasting and cheese-cheap tasting. Another improvement to consider for the current method is to deal with edge cases with negation. One example is if there is a "not" before an adjective, the "not" would be detected to prevent misleading information.

3. EXTRACT INDICATIVE ADJECTIVE PHRASES

Restaurants rely heavily on word-of-mouth and in turn good reviews by customers to improve their popularity in the market. Thus, analysing the adjectives in reviews is essential to understand the sentiment surrounding a given restaurant.

3.1 Adjective Phrases Overview

An adjective phrase is a phrase where the head is an adjective. Dependants of head adjectives are usually adverbs, prepositional phrases or clauses. There are broadly two types of Adjective Phrases - Attributive and Predicative. The structural definitions are shown below with examples:

Attributive Adjective Phrases <ADV, ADJ>

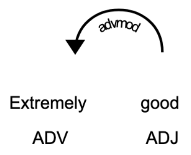


Fig 3.1 Attributive Adjective Phrase Structure

Predicative Adjective Phrases <ADJ, SCONJ, PRON>

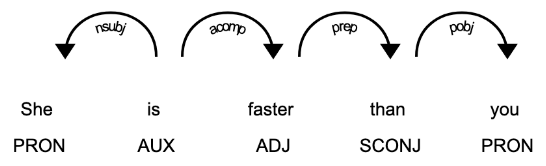


Fig 3.2 Predicative Adjective Phrase Structure

These adjective phrases are significant to gain insights into a restaurant's business. In this case, we aim to extract Indicative Adjective Phrases which uniquely identify a randomly selected restaurant b1 and summarize its characteristics.

3.2 Data Pre-processing for Reviews

Prior to the extraction and analysis of Adjective Phrases, we performed data pre-processing for all reviews. First, we performed **Segmentation** to transform the long reviews into manageable sentences and segments. Next, we performed **POS Tagging** to assign a POS tag to each word. Following these steps, we proceeded to extract the Adjective Phrases.

3.2.1 Regex for Adjective Phrases

As per the structural definition of Adjective Phrases above, we made use of Regular Expressions pattern matching to extract the required phrases from the reviews. To ensure matching for both types of adjective phrases, Attributive <ADV, ADJ> and Predicative <ADJ, SCONJ, PRON>, we defined a separate RegEx rule for each to ensure all possible cases are covered.

```
grammar1 = "ADJP: {<ADV>?<ADJ>+}"
grammar2 = "ADJP: {<ADJ>+<SCONJ><PRON>}"
```

Fig 3.3 RegEx rules for Attributive & Predicative Adjective Phrases

3.2.2 Lemmatization

Following extraction of Adjective Phrases using Regular Expressions, we performed Lemmatization to obtain the base form of the words by using morphological analysis and vocabulary. We chose lemmatization over stemming, as lemmatization reduces the inflected words properly to ensure that the root word is a valid word belonging to the language. Hence, the accuracy of base form provided by lemmatization is higher. This step ensures finding commonality between reviews for efficient grouping, consider the example below:

- "larger portion" ⟶ "large portion"
- "largest portion" ⟶ "large portion"

3.2.3 Word Embedding

Word embedding is utilized to further improve upon the results of lemmatization. Most restaurant reviews are described using common positive adjectives such as 'good', 'great' or negative adjectives such as 'bad', 'poor' and 'terrible'. These variations of positive and negative words are used to express very similar sentiments about restaurant traits such as food or service. Thus, we find the word embeddings for these adjectives to reduce a word such as 'great', 'wonderful' or 'amazing' to simply 'good'.

```
# Reference Vocabulary
good = nlp('good')
bad = nlp('bad')
threshold = 0.9

# Compute Cosine Similarity
cosine = lambda v1, v2: np.dot(v1, v2) / (np.linalg.norm(v1) * np.linalg.norm(v2))
```

Fig 3.4 Word Embedding Cosine Similarity

This is achieved using the Cosine Similarity Metric as shown above, which computes the value for a given token and compares it to a predefined threshold value (0.9 in this case) to determine the word embedding as 'good' or 'bad'. The challenges of this approach are discussed below.

3.3 Random Selection of Business b1

As required by the problem statement, a business b1 was selected at complete random as shown below:

```
unique_business = set(data['business_id'].tolist())
unique_business_list = list(unique_business)
random.seed(42)
choice = random.choice(unique_business_list)
print("Randomly selected choice: ", choice)
```

Fig 3.5 Random Selection of Business b1

Selected Business b1: `DcfkRb2bS2c8z21WH-aS6A`

3.4 Evaluation Metric: Term Frequency-Inverse Document Frequency (TF-IDF)

TFIDF is a numerical statistic that reflects how important a word is to a document in a corpus. It is computed by the product of two metrics (term frequency and inverse document frequency) where t is the term, d is the document and D is the corpus:

$$tf\,idf\,(t, d, D) = tf\,(t, d)\ .\ idf\,(t, D)$$

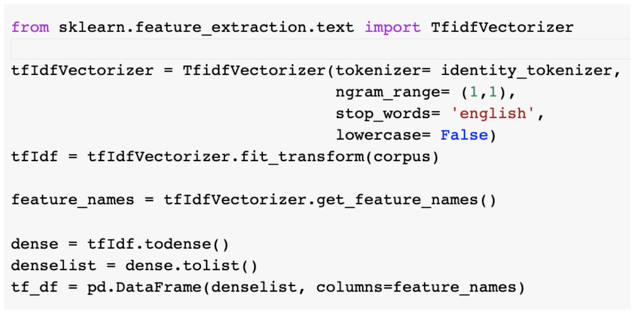
Where:

$$tf\,(t, d) = \log\,(1 + freq\,(t, d))$$

$$idf\,(t, D) = \log\,\left(\frac{N}{count\,(d \in D : t \in d)} \right)$$

Fig 3.6 TF-IDF Formula

It penalizes words that appear frequently across a corpus (inverse document frequency) and emphasizes words that are unique to a document (term frequency). Hence, in this case the aim is to maximise TFIDF for an indicative adjective phrase across all businesses other than the selected business b1.



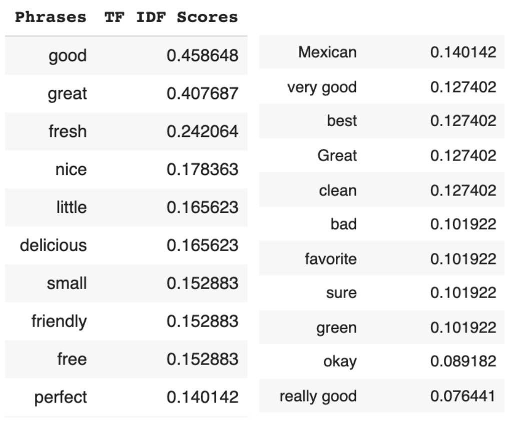
```python
from sklearn.feature_extraction.text import TfidfVectorizer

tfIdfVectorizer = TfidfVectorizer(tokenizer= identity_tokenizer,
                                  ngram_range= (1,1),
                                  stop_words= 'english',
                                  lowercase= False)
tfIdf = tfIdfVectorizer.fit_transform(corpus)

feature_names = tfIdfVectorizer.get_feature_names()

dense = tfIdf.todense()
denselist = dense.tolist()
tf_df = pd.DataFrame(denselist, columns=feature_names)
```

Fig 3.7 TF-IDF Implementation

Here the set of reviews for the selected business is treated as a document, while the entire collection of reviews from the remaining businesses are considered a corpus. Under these assumptions, sklearn's feature extraction package is utilized to perform TF-IDF analysis.

3.5 Top Indicative Adjective Phrases

After extensive analysis, the top indicative adjective phrases for selected business b1 (`DcfkRb2bS2c8z21WH-aS6A`) ranked by TF-IDF score are as follows:



Phrases	TF IDF Scores
good	0.458648
great	0.407687
fresh	0.242064
nice	0.178363
little	0.165623
delicious	0.165623
small	0.152883
friendly	0.152883
free	0.152883
perfect	0.140142
Mexican	0.140142
very good	0.127402
best	0.127402
Great	0.127402
clean	0.127402
bad	0.101922
favorite	0.101922
sure	0.101922
green	0.101922
okay	0.089182
really good	0.076441

Fig 3.8 Top Indicative Adjective Phrases

3.5.1 Manual Extraction

Manual extraction and sampling of the reviews reveals the following adjective phrases to be the most characteristic of the selected business b1 (`DcfkRb2bS2c8z21WH-aS6A`):

RESTAURANT: clean, family owned, simple, small unfancy
FOOD: delicious, fresh, authentic Mexican, flavourful, small (portion sizes), homemade, free (chips, salsa, drink refills)
SERVICE: fast, good, friendly
PRICE: great, reasonable

There are instances of some reviews that mention bad service, rude waiters and bland food - but these reviews are far fewer than the other positive reviews, and hence are treated as outliers and ignored.

3.5.2 Comparing Results

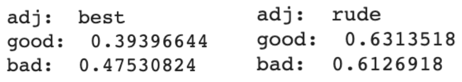
Upon comparison of manually extracted and TF-IDF ranked indicative adjective phrases, we observe a significant disparity of results. We observe that strong indicative phrases such as 'good', 'great' and 'fresh' were well captured by the TF-IDF ranking. However, other important phrases such as 'Mexican' and 'clean' are captured in the top but ranked somewhat lower than other less important phrases like 'nice' and 'perfect'. Moreover, some key indicative phrases like 'fast' and 'reasonable' are not captured by the top TF-IDF ranking at all. This may be due to the penalization of TF-IDF ranking for these phrases occurring in other businesses' reviews as well.

Hence, we can conclude that TF-IDF has produced satisfactory results by identifying some key Indicative Adjective Phrases. However, it has its limitations as it ranks some key phrases lower than it should and does not identify some phrases in the top ranking altogether. These results can be expected to skew based on phrases present in the other business reviews.

3.6 Challenges and Improvements

POS Tagging: As mentioned in the earlier sections, a token may be associated with multiple POS tags depending on the context. Hence, a challenge faced was to ensure the correct tag was assigned to each token, as incorrect tagging would affect the following steps in the pipeline.

Word Embedding: In this case, the results of word embedding were not sufficiently accurate to be useful. For instance, the adjectives 'best' and 'rude' are incorrectly classified as bad and good respectively, as shown below:



```
adj:  best                adj:  rude
good:  0.39396644         good:  0.6313518
bad:  0.47530824          bad:  0.6126918
```

Fig 3.9 Word Embedding Faults

To counteract wrong embedding, we attempted to raise the low threshold gradually to 0.9 until we reached an acceptable accuracy rate. However, the high threshold is not very useful as most word embeddings cannot match that high value. Hence, this persisted as a challenge as most words remained in their original form - rendering the process of word embedding ineffective altogether.

Evaluation Metric: We may consider an alternate metric Entropy which measures the disorganisation of a set of input values. It is high if the input values have highly varied labels, and low if many input values all have the same label. Hence, in this case the aim would be to minimize entropy for an indicative adjective phrase across all businesses other than the selected business b1.

4. APPLICATION: Sentiment Analysis

In this section, a sentiment analysis model was trained on the restaurant reviews dataset to predict the sentiment of existing reviews as well as of any other review that the user inputs.

4.1 Pre-process Data

Firstly, we needed to clean the dataset and reduce it to just reviews and their corresponding ratings. Secondly, there are 5 categories of ratings from 1 to 5. Since the size of the dataset is small, we decided to classify the ratings into three categories, namely, 0, which represents negative review, 1, which represents neutral review and 2, which represents positive reviews. We represent ratings 1 to 2 as negative, 3 as neutral and 4 to 5 as positive sentiment. Lastly, we use 80% of the data for training and 20% for testing.

4.2 Build Sentiment Analysis Model

We trained three models, Multinomial Naive Bayes (MNB), Stochastic Gradient Descent Classifier (SGDC) and Linear Support Vector Classifier (LSVC). LSVC had the best accuracy and thus, was chosen for our final application.

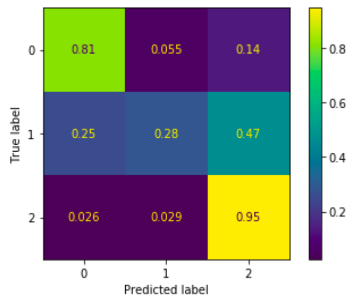
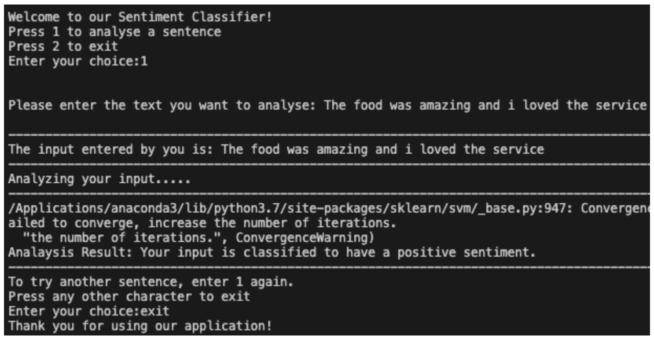


Figure 4.1 Sentiment Analysis using LSVC Model

4.3 Command Line Implementation

This model is deployed as a command line application, using which we can perform sentiment analysis on any input of the user by running the *app.py* file.

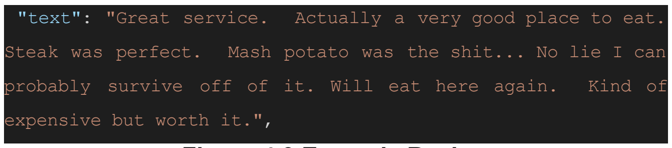


```
Welcome to our Sentiment Classifier!
Press 1 to analyse a sentence
Press 2 to exit
Enter your choice:1


Please enter the text you want to analyse: The food was amazing and i loved the service

------------------------------------------------------------------------------------------
The input entered by you is: The food was amazing and i loved the service
------------------------------------------------------------------------------------------
Analyzing your input.....
------------------------------------------------------------------------------------------
/Applications/anaconda3/lib/python3.7/site-packages/sklearn/svm/_base.py:947: Convergen
ailed to converge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)
Analaysis Result: Your input is classified to have a positive sentiment.
------------------------------------------------------------------------------------------
To try another sentence, enter 1 again.
Press any other character to exit
Enter your choice:exit
Thank you for using our application!
```

Figure 4.2 Result of sentiment analysis

4.5 Limitations

Some challenges of the sentiment analysis model include interpreting sarcasm, humour, irony, exaggeration, double negation and modern slang. For example, the term 'sh*t' is used in a positive context below, but may be misinterpreted as negative sentiment. These limitations are largely persistent in reviews due to the flexibility of expression.



```
 "text": "Great service.  Actually a very good place to eat.
Steak was perfect.  Mash potato was the shit... No lie I can
probably survive off of it. Will eat here again.  Kind of
expensive but worth it.",
```

Figure 4.3 Example Review

5. CONCLUSION

Through the process of building this rudimentary end-to-end NLP system, we discovered and employed various existing NLP toolkits which are extremely powerful for generic English-language text. However, they too have their limitations and encounter many corpus specific challenges. Hence, it may improve performance to build and use custom tokenizers/POS taggers which are designed on top of existing toolkit frameworks for the given corpus at hand. Moreover, using a statistical approach to build the custom tokenizer by training it like a machine learning model may help improve its robustness and coverage.

ACKNOWLEDGMENTS

We wish to express our sincere gratitude towards our supervisor Associate Professor Sun Aixin for his consistent support, guidance and mentorship during this project. We also wish to thank the School of Computer Science and Engineering (SCSE), Nanyang Technological University for providing us the opportunity to learn through practice by working on this project.

APPENDIX

[1] NLTK Python Library, https://www.nltk.org/
[2] https://spacy.io/
[3] https://en.wikipedia.org/wiki/Adjective_phrase
[4]https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html
[5]https://stackoverflow.com/questions/20940979/what-is-an-indexoutofrangeexception-argumentoutofrangeexception-and-how-do-i-f
[6]https://stackoverflow.com/questions/952914/how-to-make-a-flat-list-out-of-a-list-of-lists
[7]https://www.hardwarezone.com.sg/tech-news-apple-blacklists-epic-games-and-disallow-fortnite-back-app-store
[8]https://www.hardwarezone.com.sg/tech-news-m2-powered-macbook-air-rumoured-launch-q3-2022
[9]https://www.channelnewsasia.com/world/australia-covid-19-sydney-unvaccinated-lockdown-freedom-2206346
[10]https://www.channelnewsasia.com/singapore/urban-farms-pesticide-free-vegetables-accreditation-scheme-singapore-2204351