Module 3 Report

XINKAI CHEN, WENBO FEI, GUANQI LU, LOKESWAR SADASIVUNI

1.Introduction

1.1 Abstract

In this project, we used dataset from Yelp. We focused on the Chinese restaurants in three states in Canada: Quebec, Alberta, and Ontario. There are 1370 restaurants in total.The goal of our project is to analyze the influence of five aspects on the ratings and to provide suggestions for the restaurants in our dataset based on the aspect category ratings we explored. The categories are food, service, ambience, price and miscellaneous.

1.2 Data Background

The Yelp data contains 4 json files, **review.json** contains information of 6,685,900 reviews, **business.json** contains information of 192,609 businesses, **user.json** contains information of 1,637,138 users, and the **tip.json** contains information of 1,223,094 tips.

1.3 Data Selection

To make sure that both the results are resonable and there are enough restaurants to study, we only keep the restraunts that have more than 10 reviews. At last we have 1370 restaurants and 70506 reviews.

```
In [2]: import pandas as pd
        business=pd.read_csv("chinese_canada_review10.csv")
        business.head()
```

Out[2]:

	address	attributes	business_id	categories	city	hours	is_open	latitu
0	503 College Street	{'BestNights': "{'monday': False, 'tuesday': F...	_5XClj4E5VClsEscbrrPKg	Bars, Nightlife, Chinese, Restaurants, Cocktai...	Toronto	{'Tuesday': '18:0-2:0', 'Wednesday': '18:0-2:0...	1	43.6558
1	469 Queen Street W	{'BikeParking': 'True', 'RestaurantsTakeOut': ...	_AiqOS8io_reYZri1OeP1g	Restaurants, Food, Chinese, Bubble Tea	Toronto	{'Wednesday': '12:0-19:0', 'Thursday': '12:0-1...	0	43.6483
2	100 King St W	{'RestaurantsGoodForGroups': 'True', 'Restaura...	_fe36eepp9tsP5vzvKbdQA	Food, Restaurants, Bakeries, Chinese	Toronto	NaN	1	43.6486
3	4188 Finch Avenue E, Unit14-15	{'BusinessParking': "{'garage': False, 'street...	_GwVhLhVF_NkEjdcY0TZ8w	Asian Fusion, Barbeque, Restaurants, Chinese	Toronto	{'Monday': '15:30-1:0', 'Tuesday': '15:30-1:0'...	0	43.8039
4	643 Bloor Street W	{'Ambience': "{'romantic': False, 'intimate': ...	_HePiPCk9tmgezshWwZGOw	Chinese, Restaurants	Toronto	{'Monday': '11:0-22:0', 'Wednesday': '11:0-22:...	1	43.6642

2. Aspect Based Sentiment Analysis (ABSA)

Sentiment analysis is currently widely used as one important method of Neural Lauguage Processing (NLP) both in the industry and academics. However, the majority of traditional approaches more tends to detect the overall sentiment polarity of a sentence, paragraph or text span, regardless of the entities mentioned (also regarded as aspect category, e.g. food, service) and their aspect terms (e.g. noodle, burger; waiter, staff). On the controary, this task is focusing on Aspect Based Sentiment Analysis (ABSA), where the goal is to identify the aspect terms of given target entities and the sentiment expressed towards each aspect terms. [1] To complete the whole analysis, we will firstly pre-process the review data and go through 4 basic subtasks for ABSA.

2.1 Review Data Pre-Processing

Before we begin the aspect based sentiment analysis, we must clean the raw reviews. Given our reviews are coming from Canada, some of them are unavoidably written by French. Therefore the first step is to find all non-English reviews and remove them, since our report only concerns about the English NLP. The next step is to replace all pronouns showing in the reviews by using a pre-trained neural coreference model. Some of pronouns might correspond to important aspect terms, which the extraction process will not find out since

```
In [5]:  import neuralcoref
         import spacy
         nlp = spacy.load('en_core_web_lg')
         neuralcoref.add_to_pipe(nlp)
         doc= nlp("I tried the chicken for lunch, but it tasted awful.")
         doc._.coref_resolved

Out[5]:  'I tried the chicken for lunch, but the chicken tasted awful.'
```

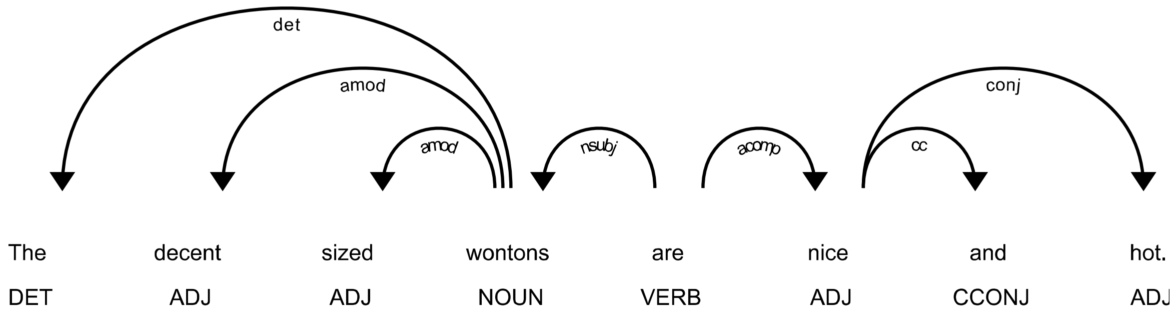
This method will replace "it" with "the chicken". The last review pre-processsing part is to split the large chunk of reviews into multiple sentences, which will make our aspect term extraction task much more easier.

2.2 Task 1&2: Extracting Aspect Terms and Computing Sentiment Scores

Aspect terms are the objects which one review will mention most intuitively, for instance, "the chicken" in last example shown above. It's also what we most concern in the ABSA. Looking for the aspect terms directly will be a little bit hard, since we don't know what exact aspect term that one review could have. A common solution, by contrast, is to firstly find out the opinions words from the sentence given two lexicons of positive words and negative words[2][3], then use the spaCy dependency tree[4] to search the particular aspect terms that these opinion words correspond to. This allows us to extract the aspect term easily, which is showing in the image and the list below.

```
In [12]: import spacy
         from spacy import displacy
         nlp = spacy.load("en_core_web_lg")
         doc = nlp("The decent sized wontons are nice and hot.")
         displacy.render(doc, style="dep",jupyter=True,options={"distance":100})

Out[12]:
```



```
In [11]: for token in doc:
             print(token.text, token.dep_, token.head.text, token.head.pos_,
                     [child for child in token.children])

         The det wontons NOUN []
         decent amod wontons NOUN []
         sized amod wontons NOUN []
         wontons nsubj are VERB [The, decent, sized]
         are ROOT are VERB [wontons, nice, .]
         nice acomp are VERB [and, hot]
         and cc nice ADJ []
         hot conj nice ADJ []
         . punct are VERB []
```

By searching from the children of the VERB "are" and the NOUN "wontons" itself, we can find the opinion words like "nice", "hot" and "decent", which are corresponding to "wontons", the aspect term we expect. For more specific extracting process, you can find it in our codes. Then, we define a set of rules to set the correct sentiment score to the opinion word (i.e. flipping the sign of the sentiment when negation words are present), and assign that sentiment score to the aspect term that it's referring to. There are three basic rules of assigning the sentiment scores:

1. If the aspect term has to one positive opinion word, its sentiment score will be plus 1.Vice versa.
2. If the opinion word has an intensifier like "very" and "extremely", the sentiment score of it will be mutiplied by 1.5.
3. If two opinion words with same polarity showing up in the same sentence and connected by the conjunction like "and", their sentiment score will not be plus 2 or minus 2, but plus or minus 1.5 for the penalty of too many sentiment words.

```
In [15]: from extract2 import get_senti
         senti_score={}
         opi={}
         review="This was my first time here and i decided to order the shrimp wonton soup. The soup itself
         was basic and came with 4 decent sized wontons, nice and hot. They also have a self serve tea sect
         ion.The service was fast. Im talking about placing our order and within 1 min my soup was on the t
         able!"
         get_senti(review,senti_score,opi)
         senti_score

Out[15]: {'wonton': 2.5, 'service': 1}
```

```
In [16]: opi

Out[16]: {'wonton': ['decent', 'nice', 'hot'], 'service': ['fast']}
```

The example above shows the output result of the aspect term extracting and sentiment polarity scores for one review.

2.3 Task 3&4: Aspect Category Assignment and Calculating Category Scores

By summerizing the results from our preliminary analysis, we determine five different aspect categories in total: "Food", "Service", "Ambience", "Price" and "Miscellaneous". The business owners can find which aspect they can improve for their restaurants from the sentiment score of each category. After we go through all the reviews, we already get hundreds of aspect terms aligning with their sentiment score for each business. The next task is to assign the aspect terms to the aspect category they belong to. For example "wonton" belongs to "Food" and "service" belongs to "Service". The plan is to use the word similarity test from WordNet package to compute four similarity scores between one aspect term and four category words, "Food", "Service", "Ambience" and "Price". This aspect term will be assigned to the category which has the highest similarity with it. If all four similarity scores are significant small, then this aspect term will be assigned to the "Miscellaneous" otherwise. The similiarity score between "wonton" and "food" is 0.769, much higher then the other three, therefore it will be assigned to the "Food" correctly. In the same time the "Food" category will get both the aspect term "wonton" and its sentiment score 2.5 under it.

In [23]:

```
from nltk.corpus import wordnet
from itertools import product
sense1 = wordnet.synsets("wonton")
sense2 = wordnet.synsets("food")
inner = []
for s1, s2 in product(sense1, sense2):
    score = wordnet.wup_similarity(s1, s2)
    inner.append(score)
max(inner)
```

Out[23]:

```
0.7692307692307693
```

Finally we just sum all the positive sentiment scores and all the negative scores seperately under each category. These will be the raw sentiment scores for each category in our analysis. The head of our output result is showing below.

In [26]:

```
import pandas
df=pandas.read_csv(r'score_output2.csv', encoding = "ISO-8859-1")
df.head()
```

Out[26]:

	business_id	food.pos	food.neg	service.pos	service.neg	ambience.pos	ambience.neg	price.pos	pric
0	_5XClj4E5VCIsEscbrrPKg	59.50	-16.5	47.0	-24.00	20.50	-1.0	1.0	
1	_AiqOS8io_reYZri1OeP1g	72.00	-13.5	25.0	-12.00	14.75	0.0	6.0	
2	_fe36eepp9tsP5vzvKbdQA	39.75	-9.0	24.0	-6.75	9.00	-4.0	4.5	
3	_GwVhLhVF_NkEjdcY0TZ8w	47.50	-19.0	48.5	-17.50	11.75	-6.0	3.5	
4	_HePiPCk9tmgezshWwZGOw	40.75	-24.0	35.0	-14.50	19.75	-4.0	4.0	

3. Predicton and Model Evaluation

3.1 Read Data：Sentimental Score

using this to run R : %load_ext rpy2.ipython,if haven't install rpy2, please do it.

In [3]:

```
%load_ext rpy2.ipython
```

```
The rpy2.ipython extension is already loaded. To reload it, use:
  %reload_ext rpy2.ipython
```

In [4]:

```
%%R
rawdata = read.csv("score_output2_normalize+review count+stars.csv")
colnames(rawdata)
```

Out[4]:

```
 [1] "business_id"   "food.pos"      "food.neg"      "service.pos"
 [5] "service.neg"   "ambience.pos"  "ambience.neg"  "price.pos"
 [9] "price.neg"     "misc.pos"      "misc.neg"      "review.counts"
[13] "star"          "fp"            "fn"            "sp"
[17] "sn"            "ap"            "an"            "pp"
[21] "pn"            "mp"            "mn"
```

In [5]:

```
%%R
#explain each colnames, like fp=food.pos/review.counts, fn=food.neg/review.counts, food.pos, food.
neg is generated from sentimental analysis
table(rawdata$star) # Yelp review score, "discrete", imbalanced"
```

Out[5]:

```
  1 1.5   2 2.5   3 3.5   4 4.5   5
  2   4  34 130 395 497 251  56   1
```

In [7]:

```
%%R
rawdata$response = rawdata$star >=3.5 # take median as threshold, transform into a binary variable
(good or bad)
```

3.2 Prediction

3.2.1 Find weight of positive reviews v.s negative reviews

At first, fit a linear model of these 10 sentimental scores directly from our analysis

In [8]:

```
%%R
model.adj = glm(response ~ fp + fn + sp + sn + ap + an + pp + pn + mp + mn,
                family = binomial,  data = rawdata)
summary(model.adj)$coefficients
```

Out[8]:

```
              Estimate Std. Error  z value     Pr(>|z|)
(Intercept) -0.3607774  0.2853498 -1.264334 2.061102e-01
fp           0.6752522  0.1113185  6.065950 1.311763e-09
fn           0.9211210  0.1678120  5.489007 4.042004e-08
sp           0.3829438  0.1214519  3.153050 1.615744e-03
sn           1.9295200  0.2088881  9.237100 2.532688e-20
ap           0.5879515  0.2696129  2.180725 2.920378e-02
an           0.6557280  0.3598291  1.822332 6.840466e-02
pp           2.0035592  0.4080062  4.910609 9.079384e-07
pn           1.5799069  0.6983734  2.262267 2.368093e-02
mp           1.5878011  0.4607829  3.445877 5.692103e-04
mn           1.2899643  0.7885253  1.635920 1.018563e-01
```

3.2.2 calculate score of each category

Take category "Food" as example, for other 4 categories, the process is the same. Since positive reviews and negative reviews have different effect on determine whether a restaurant is good or not, using weighted proportion of positive sentimental score to evaluate the category,

$$\texttt{adj.fp} = \frac{\omega_{f.pos} * \texttt{fp}}{\omega_{f.pos} * \texttt{fp} + \omega_{f.neg} * \texttt{fn}}.$$

Where $\omega_{f.pos} = \frac{\beta_{fp}}{\beta_{fp}+\beta_{fn}}$, β_{fp}, β_{fp} are the regression coefficients of the `model.adj`. Then normalize it into 0-5 scale as our final score for the category.

In [9]:

```
%%R
rawdata$adj.fp = model.adj$coefficients[2]*rawdata$food.pos/(model.adj$coefficients[2]*rawdata$foo
d.pos-model.adj$coefficients[3]*rawdata$food.neg)
#negative score is always negative, so we take absolute (using minus sign)
rawdata$norm.fp = ((rawdata$adj.fp-min(rawdata$adj.fp))/(max(rawdata$adj.fp)-min(rawdata$adj.fp)))
*5
```

3.3 Evaluation and Interpretation

Use cross validation to see whether the scores of 5 categories generated from sentimental analysis on Yelp reivews can really predict whether a bussiness is good or not.

In [11]:

```
%%R
score = read.csv("cat_rating_raw.csv")
library(boot)
cost <- function(r, pi) mean(abs(r-pi) > 0.5)
model1 = glm(response ~ norm.fp + norm.sp + norm.ap + norm.pp + norm.mp ,
             family = binomial, weights = review.counts, data = score[,c("norm.fp","norm.sp","nor
m.ap","norm.pp","norm.mp","response", "review.counts")])
summary(model1)$coefficients
```

Out[11]:

```
              Estimate Std. Error  z value      Pr(>|z|)
(Intercept) -9.1808746 0.10347294 -88.72730  0.000000e+00
norm.fp      0.9770836 0.01869698  52.25889  0.000000e+00
norm.sp      1.1346190 0.02590849  43.79333  0.000000e+00
norm.ap      0.3101540 0.01941195  15.97748  1.834158e-57
norm.pp      0.5368046 0.01732641  30.98187 9.459207e-211
norm.mp      0.6476284 0.01726579  37.50934 6.486253e-308
```

From the p-value, we can see that all the 5 categories are significant when we want to determine whether a restaurant is good or not. All of the coefficients are positive, so the positive score of these five aspects will indicate a restaurant is good. The food and service aspects' coefficients id larger, so they are the most important among the five.

```
In [0]: %%R
        cv1 = cv.glm(score[,c("norm.fp","norm.sp","norm.ap","norm.pp","norm.mp","response", "review.count
        s")], model1, cost)
        1-cv1$delta[1] ###[1] 0.7525547
```

The Leave-one-out cross validation shows our whole analysis process would achieve a 75% accuracy predicting whether a restaurant is good or not based on the review data.

4. Shiny APP

Please check our shiny App: https://ericchenzhang.shinyapps.io/yelp/ (https://ericchenzhang.shinyapps.io/yelp/)

5. Conclusion

From our anaysis we can find out that among the five aspects(food, service, ambience, price and miscellaneou), the food and service are the two aspects that affect the rating of Chinese restaurants the most. As a result, business owners are supposed to pay more attention to the food and service.
Our project provided the business owners with the scores of their restaurants in five aspects, also comparing them with the restaurants in the same region and same kind of restaurants. According to the scores and comprisons they will be able to see which aspect is better than other restaurants and which aspects need to be improved.
Due to the time constrain, we are not able to dig into the attributes of business dataset, in future study, this may take into account.

References

[1] Aspect Based Sentiment Analysis(ABSA) https://github.com/SaujanyaReddy/Aspect-Based-Sentiment-Analysis-IRE-Major-Project (https://github.com/SaujanyaReddy/Aspect-Based-Sentiment-Analysis-IRE-Major-Project)
[2] Aspect-Based Opinion Mining (NLP with Python) https://medium.com/@pmin91/aspect-based-opinion-mining-nlp-with-python-a53eb4752800 (https://medium.com/@pmin91/aspect-based-opinion-mining-nlp-with-python-a53eb4752800)
[3] Opinion lexicon https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html (https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html)
[4] spaCy Usage Documentation https://spacy.io/usage/linguistic-features (https://spacy.io/usage/linguistic-features)

Contributions

Ideas are discussed and determined by all of us
Xinkai Chen: Sentiment analysis, shiny app, and final summary.
Wenbo Fei: Prediction and model evaluation, slides, and final summary.
Guanqi Lu: Data pre-processing and cleaning, final summary, and organizing Github repository.
LOKESWAR SADASIVUNI:Category selection, cusine categories, and slides.