Positive Unlabelled Learning
to Recognize Dishes as Named Entity

التعلم من الأمثلة الصحيحة والمجهولة
للتعرف على الأطباق ككيان مسمى

by

AIMAN TAREK

Dissertation submitted in fulfilment

of the requirements for the degree of

MSc INFORMATION TECHNOLOGY MANAGEMENT

at

The British University in Dubai

April 2019

DECLARATION

I warrant that the content of this research is the direct result of my own work and that any use made in it of published or unpublished copyright material falls within the limits permitted by international copyright conventions.

I understand that a copy of my research will be deposited in the University Library for permanent retention.

I hereby agree that the material mentioned above for which I am author and copyright holder may be copied and distributed by The British University in Dubai for the purposes of research, private study or education and that The British University in Dubai may recover from purchasers the costs incurred in such copying and distribution, where appropriate.

I understand that The British University in Dubai may make a digital copy available in the institutional repository.

I understand that I may apply to the University to retain the right to withhold or to restrict access to my thesis for a period which shall not normally exceed four calendar years from the congregation at which the degree is conferred, the length of the period to be specified in the application, together with the precise reasons for making that application.

Aiman Tarek

_____

Signature of the student

COPYRIGHT AND INFORMATION TO USERS

Abstract

With the growth of social media, there is a need to analyse the user-generated content; especially the text reviews. Online text reviews have a lot of potential and opportunities for both users and business owners. Many researches target analysing text reviews extracting useful info especially Named Entity Recognition.

In this research, I focus on extracting food and dish names as a named entity. With the lack of labelled data, I try to overcome the cold start and avoid manual labelling by building a lookup table from a dictionary.

I work with Yelp dataset, going through each text review, using each noun as a candidate, label the positive samples using the aforementioned lookup table, then using Positive Unlabelled learning techniques to recognise more entities within the unlabelled data, by predicting the probability for each candidate. I considered the surrounding words; preceding and following in building the model, as well as Part of Speech tag for each. To eliminate duplicates due to repeated candidates from different reviews or sentences, I calculate the average and represent each candidate entity only once.

The results show how we can automate entity recognition process, using dictionaries and machine learning techniques and achieve an acceptable accuracy of 67% and boost the newly discovered entities by around 15% using Positive Unlabelled learning over automatically build lookup table.

This research has the potential to be extended to other topics other than food and dish names, also it acts as a framework and algorithm independent.

الخلاصة

مع النمو المتصاعد لشبكات التواصل الاجتماعي، ظهرت الحاجة إلى تحليل المحتوى المنشور بواسطة المستخدمين؛ خصوصا الآراء النصية. تتيح الآراء النصية الكثير من الإمكانيات والفرص لكل من المستخدمين وأصحاب الشركات. ركزت الكثير من الأبحاث على تحليل الآراء النصية لاستخراج المعلومات المفيدة منها، خصوصا التعرف على الكلمات ككيان مسمى.

في هذا البحث، أركز على استخراج أسماء الأكلات والأطباق ككيان مسمى. بسبب نقص البيانات السابق تعريفها، أحاول التغلب على البدايات البطيئة وتجنب اللجوء إلى التعريفات اليدوية عن طريق بناء جدول تحقق باستخدام القاموس.

استخدمت بيانات يلب، مرورا بكل رأي نصي، واختيار الأسماء كمرشح، مع تعريف الأمثلة الصحية باستخدام جدول التحقق السابق ذكره، ثم استخدام طرق التعلم من الأمثلة الصحيحة والمجهولة للتعرف على كيانات أكثر من الأمثلة المجهولة عن طريق توقع احتمال كون كل اسم مرشح كذلك. أخذت في الاعتبار الكلمات المحيطة؛ سواء كانت سابقة أم تالية، وأيضا توصيف كل كلمة حسب وقوعها في سياق الكلام. للتخلص من التكرارات المتشابهة نتيجة وجود مرشح أكثر من مرة من آراء أو جمل مختلفة، قمت بحساب المتوسط والتعبير عن كل مرشح مرة واحدة فقط.

أظهرت النتائج مقدرتنا على ميكنة عملية التعرف على الكيان المسمى، باستخدام القاموس وأساليب تعلم الآلة، وحققت دقة مقبولة تصل إلى 67% ودفع الأسماء الجديدة المكتشفة باستخدام طريقة التعلم عن طريق الأمثلة الصحيحة والمجهولة بنسبة 15% مقارنة مع استخدام جدول التحقق فقط.

هذا البحث يمكن تمديده إلى مجالات أخرى غير أسماء الأكلات والأطباق، بل هو نمط عمل ويوفر المرونة في استخدام لوغاريتمات أخرى.

Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	Definition
NER	Named Entity Recognition
PU Learning	Positive Unlabelled Learning
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
PoS	Part of Speech
ROI	Return on Investment

Chapter 1: Introduction

With the rise of social media and user-generated content, online reviews became an essential part of our daily life. Before buying any product or service we check the reviews and compare pros and cons of each one, thus, reviews help in narrowing down the options and overcoming the hassle of choice. Text reviews support decision making and help in providing recommendations to users especially in the tourism industry (Rossetti, Stella, & Zanker, 2016).

A review can give users honest feedback and good recommendations. A good review drives and supports consumers' purchase decisions (S. Lee & Ro, 2016). For business owners, reviews provide insights about a product or a service. Online reviews help them grow and predict market trends (Morris & Edalat, 2015). Also, analysing user reviews can help recommender systems to learn from user preferences and provide better choices based on previous experience.

After realising the importance and power of social media, many online businesses started as a space or a platform for users to express their opinions or share their experiences. Yelp includes over 177M reviews with around 164M monthly unique users (Yelp, 2019) valued at $ 2.9B (Bloomberg, 2019). TripAdvisor has over 730M reviews with over 490M monthly unique visitors (TripAdvisor, 2019) with a market cap of $ 6.3B (Bloomberg, 2019).

Since restaurants reviews make-up a huge chunk of Yelp; in this research, I will focus on analysing restaurant reviews. I will try to extract food names and recognize dishes as a named entity.

The challenge of free-text reviews in restaurants domain comes mainly from having multi-aspect unstructured text, it includes a biased opinion of some positive and negative aspects, it reflects some feelings and emotions as well. It includes some dish names which could be coupled by an opinion.

Review	Source
"I loved the pasta here! It was so good and flavorful. It was also filling. It came with bread and a salad. I have also had the pizza here which was great also. A little bit of slow service around lunchtime, but well worth it."	Abby on Yelp
"I tried this place as i had an entertainer voucher ..loved my sizzling chicken and the brownie with vanilla ice cream..the only disappointment was the staff wasn't very welcoming also the service was really slow..over all we enjoyed our meal."	Rahila on Zomato.

Table 1: Food text reviews sample

By analysing the reviews, I find that the reviewer used some food and dish names "in light blue": pasta, bread, salad, pizza, sizzling chicken, brownie, vanilla ice cream. They mentioned some other aspects "in dark blue": voucher, service and staff. For what they liked, the reviewers used positive keywords "in green": loved, good, flavourful, filling, great, worth and enjoyed. While they used some negative keywords "in red": slow, disappointment, not very welcoming and slow.

Over time, it became impossible to read all the reviews due to a large number of reviews. Hence, we have a need for automating the process of information extraction and providing only the relevant piece of information.

If we want to build a system that would recommend a place, we can check highest rated restaurants, however recommending a dish requires going much deeper into reviews, extracting dish names and linking it to its valence; positive or negative.

A lot of effort has been directed towards making the most out of the online reviews, especially recommendation systems (Rossetti et al., 2016); some basic systems suggest the most common items powered by other users' choices. Not only analysing the review as a whole, but going deeper to sentence level and even better to each aspect of the review (Chinsha & Joseph, 2014), and linking each aspect with a sentiment.

Most of the researches in the field of opinion-mining aim to extract some information from of a huge amount of natural language unstructured text, and sometimes take an action based on this info (Ren, El-Kishky, Wang, & Han, 2016). However, faced by the absence of manually annotated data for a specific domain, it is not easy to use conventional machine learning techniques to identify entities or build a corpus.

One of the most important steps in opinion or reviews mining is Named Entity Recognition (NER); where researches aim to identify some items of interest, then link them with a sentiment analysis's result that indicates whether this item has a positive, neutral or negative opinion. General domain entity recognition pre-trained models aren't sufficient for domain-specific corpora (Ren et al., 2016). Hence there is a need to build a domain-specific text corpus.

To solve the challenge of having an unlabelled dataset with some known positive samples, and no negative samples, Positive Unlabelled Learning (PU Learning) is used (Mordelet & Vert, 2014); similar to building a machine learning model from the semi-supervised dataset, sometimes referred to as one-class classification.

Entity recognition is used for multi-aspect reviews, for example, if you are trying to find a hotel on Booking.com, you will find keyword filters that would give you suggestions like couple-friendly, or something about the swimming pool. NER is used in online advertisements as well, for Google and Bing Ads (Ren et al., 2016), to provide users with relevant advertisements based on the web page content or search keywords.

1.1 Problem Definition

Automating the discovery of dish names from text reviews comes with many challenges. First, there is no specific structure for the review; it's a free-text coming from users from multiple cultures and backgrounds, some of the reviews are written by non-native English speakers. The ever-increasing amount of user-generated content which makes it impossible to manually read or analyse text reviews. Also, there is no single dictionary that includes all the food or dish names that are out there (Chinsha & Joseph, 2014).

Hence, there is a need to build a system that can overcome the challenge of the named-entity discovery process with minimal human effort, not only in the food industry but almost in every aspect of the business, operating online or on social media, with user-generated reviews.

1.2 Existing systems

Many systems have been developed with NER in mind, they can achieve a certain level of analysis, yet not that extensive, mainly for open domain entity recognition. (Dale, 2018) explores the bigger players in text analysis. **Amazon** motivated by Alexa needs, their smart digital assistant, which resulted in Amazon comprehend (Amazon, 2018). **Google** has Natural Language API which has entity recognition capabilities among other text analysis tools. **IBM** has one of the best text analysis platforms; their Natural

Language Understanding which excels in topic recognition and categorizing content with multi-levels. **Microsoft** has their NLP tools coupled with Cognitive services and provides some of the standard text analysis along with language detection and linking entities. Also, **Stanford** has CoreNLP which provides core-technology for those who want to build their own platforms.

1.3 Research Objective and Questions

The research goal is to build a system that can recognize food and dish names as a named entity without the need for the human labelled dataset.

As a result of the above I ended up having the following research question which I aim to answer:

- Can I analyse free text review to recognize dishes and food names?
- How to get over the cold-start of the completely unlabelled Yelp dataset of and the absence of a corpus for food?
- Which features to consider tackling NER for dishes in the restaurant reviews?
- Is PU learning effective in recognizing dishes as a named entity?

1.4 Proposed Solution

A system that acts as a framework to automate the NER process without the need for human intervention. The framework consists of two stages; the first one is building a dataset and automatically labelling it using a lookup dictionary (Ren et al., 2016), the second stage is using machine learning techniques to build a model that can predict the probability of each word in the dataset to be a named entity using PU learning techniques.

1.5 Main Contributions

This research contributes to solving the challenges of NER for dishes in restaurants text reviews. First, I overcame the cold-start by building a lookup table using a dictionary. Then I considered four words before and after each candidate, including the words PoS tags. The research proved how effective is it to use PU learning techniques in tackling NER in general and especially for dishes. I could achieve 67% accuracy for NER using PU learning. Also, the system improved NER by 15%, using machine learning techniques compared to using a dictionary-generated lookup table.

1.6 Document Structure

The dissertation document is divided into 6 chapters. Chapter 2 covers literature review to the related topics, chapter 3 introduces the research method steps in detail, chapter 4 is the results of the research, in chapter 5 I discuss the findings, chapter 6 is the conclusion.

Chapter 2: Literature Review

In this chapter, I cover four main topics, starting by the importance of social media and user-generated content, then Named Entity Recognition (NER), Positive Unlabelled (PU) learning, and some researches related to Yelp dataset.

2.1 The Power of Social Media

User-generated content includes loads of useful info that would contribute to decision making for business owners and affects the choices and purchase decisions for consumers as well.

Online consumer reviews express personal experiences, (Zhang, Ye, Law, & Li, 2010) explore the effect of positive reviews on the restaurants in terms of generating hits on their webpage, which proven to increase the restaurant popularity.

(S. Lee & Ro, 2016) discuss how online reviews can change the attitude of the consumer towards a certain business, they discuss how positive and negative reviews would affect the consumer preference, they proved how reading a negative review of a business or a product would alter the consumer decision more than a positive one.

Online sources and social media offered a great source of recommendations and support of choices, hence we have a need for social media mining, (Rossetti et al., 2016) explore topic modelling for reviews especially in the tourism industry, they try to extract information with a specific theme.

(Morris & Edalat, 2015) indicate how the info gathered from social media can boost the restaurant revenue by profiling the guests, targeting certain customers and addressing

their needs. They mention how having loyal customers would help in building a positive online reputation and investing in social media has a good Return on Investment (ROI).

(Kim, Li, & Brymer, 2016) proved that the number of social media reviews have a positive effect on the restaurant performance in general and it has an influence on increasing the sales, attract more guests to the restaurant and it will lead to relatively higher spends.

2.2 Named Entity Recognition

The named entity recognition techniques have evolved from old rule-based techniques to heuristic methods, to machine learning methods (Nadeau & Sekine, 2007). Rule-based techniques include using fixed lexical rules and regular expressions (Brin, 1998) cited in (Nadeau & Sekine, 2007), and monitoring the upper case and lower case words to identify candidates too. (Chao, Chu, Ho, Wang, & Tsai, 2016) introduced a framework that starts by extracting dish names from Yelp dataset using regular expressions.

In the case of unlabelled data, using the WordNet dictionary to give a label to input and compare it to a list of a specific context in a form of topic modelling (Alfonseca & Manandhar, 2002). Semi-supervised learning is recently used for NER, the technique used is known as “bootstrapping”, it’s good to overcome the cold start; when you have a small set of positive examples. The method tries to find names that come in a similar context; this method requires large numbers of samples, however, the performance can exceed supervised methods (Nadeau, Turney, & Matwin, 2006).

The larger sample size only isn’t enough; including some similarity measures and choosing more relevant samples can help to improve the results (Ji & Grishman, 2006). They indicate that bootstrapping is used to enhance NER with unlabelled data as well,

and identifying a pattern with part of speech tags is proven effective (Collins & Singer, 1999).

For open domain entity recognition, (Bowden, Wu, Oraby, Misra, & Walker, 2018) built a system, named Slugbot's Named Entity Recognition for dialogue Systems (SlugNERDS) based on Google Knowledge Graph Search API, which uses Schema.org implicitly.

(Ren et al., 2016) indicate why we cannot use existing and pre-trained entity recognition model, that was built for a general domain, on domain-specific corpora, they also mention how challenging is the ambiguity of entities depending on the context. Most of the generally available NER algorithms are built on long text documents, so they perform poorly on the shorter text like social media comments, tweet and online reviews (Bontcheva, Derczynski, & Roberts, 2017).

2.3 Positive Unlabelled Learning

The conventional machine learning techniques are either supervised techniques from labelled positive and negative examples or unsupervised techniques from unlabelled samples. However, the Positive Unlabelled (PU) Learning is different since it includes positive and unlabelled examples, with an absence of negative examples; unlabelled can fall under either a positive or a negative category.

In general solving PU learning problems were tackled by a naïve solution of considering the positive samples as positive samples as true, negative as false, and try to predict the probability and chances of the unlabelled examples of being positive or true (Sansone, De Natale, & Zhou, 2018). Others introduced two steps techniques (Kaboutari, Bagherzadeh, & Kheradmand, 2014) which involved finding negative samples in the

unlabelled data, then apply the conventional machine learning techniques (X.-L. Li & Liu, 2005; Wright, 2017).

Also, some novel techniques were introduced like Augmented EM (X.-L. Li & Liu, 2005) which is based on considering positive samples as noise, then generating a sequence of classifiers and selecting a good classifier, based on Naïve Bayesian classification. Weighted Logistic Regression is used by (W. S. Lee & Liu, 2003) to deal with high dimensionality. Also (C. Li & Hua, 2014) introduced a Positive Unlabelled Random Forrest (PURF) which achieves better parallel performance and uses bootstrapping with replacement. (Elkan & Noto, 2008) introduced a method that starts by randomly labelling some of the unlabelled data as positive, then train two classifiers; one using all the data, and another one using all the unlabelled examples with 20% of the positive ones. (Sansone et al., 2018) proposed Unlabelled data in Sequential Minimal Optimization (USMO) which mainly solves the issue of limited resources by dividing it to smaller subsets. One of the best techniques introduced by (Mordelet & Vert, 2014) which is bagging, to build the model using a sample of positive examples and a random sample of similar size of the unlabelled data, then score the model against the remaining of the unlabelled data, by shuffling the unlabelled samples we end up with a score of the probability of unlabelled data being positive. Bootstrapping is proven effective in semi-supervised learning and unlabelled data (Gupta & Manning, 2014; Lin, Yangarber, & Grishman, 2003).

PU learning is used when faced with a huge dataset which includes a very small portion of known positive examples and the majority is unlabelled (Claesen, De Smet, Suykens, & De Moor, 2015), some of the conventional approaches are affected by the percentage of positive examples (Skabar, 2002). They introduced a novel technique based on SVM.

Some algorithms like modified Naïve Bayes can be used to solve PU but we should know the probability of the positive sample (W. S. Lee & Liu, 2003).

2.4 Yelp Dataset

Searching for the word "Yelp" on google scholar leads to over 41,200 search results. With around 1,340 of them about "Yelp Dataset"

(Huang, Rogers, & Joo, 2014) used the Yelp dataset to discover hidden topics, to give restaurants some insights and direct them towards customer concerns.

Some other researches focused on using text reviews to predict the star rating for a business. (Fan & Khademi, 2014) managed to do so, by extracting most common features. While faced by the challenge of the inconsistency of user reviews compared to professionals, they could predict the rating by considering biased opinions and extracting positive and negative aspects.

(Zhao, Han, Meng, He, & Zhang, 2017) emphasise the power of websites like Yelp, they explain how the users' count is significantly higher than items, which provide such rich resource for info with thousands of info. They learn user preferences from reviews by doing semantic analysis, then offer item recommendation based on ratings, previous choices, and other users' ratings.

Chapter 3: Research Method

In this chapter, I introduce a framework that tackles all the stages of NER; starting with free text reviews, I give a brief introduction about the database I have. Then I show how to overcome the cold start of completely unlabelled data. Then transforming the free-text into a proper dataset that matches our needs. Then I build the model using PU learning techniques, and finally postprocessing the results.

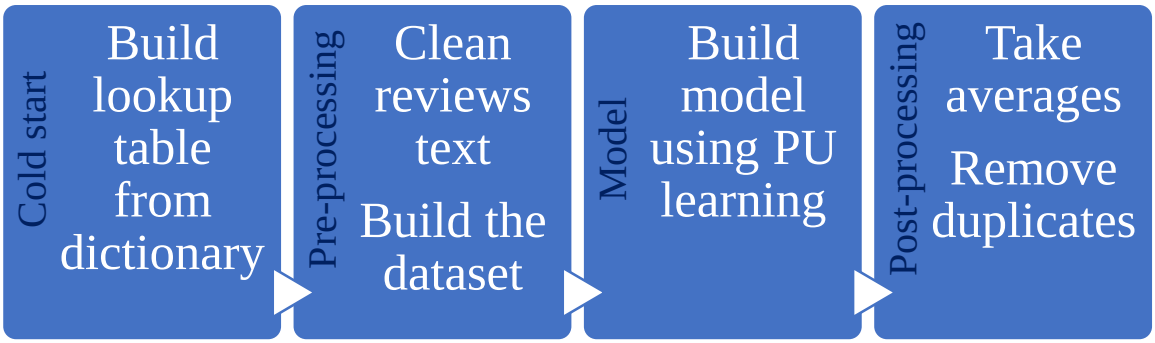

Figure 1: Research method overview

Yelp is a crowd-sourced website that has a huge traffic of over the 34M monthly unique visitor. It has more than 177 million reviews as of Q3 2018. To give you a general idea about the Yelp dataset (Yelp, 2019), it includes 6.7M reviews, each review with the following fields:

- id
- business_id
- user_id
- stars
- date
- text "includes the review text"
- useful
- funny
- cool

id	--AmbUiukqrLJpZSSjXO0g
business_id	ztmIXLuIAADzuTa4skUTgg
user_id	lu5Gl35WhpqIkg06o9NaJw
stars	5

date	2016-09-05 00:00:00
text	Becker's has actually spoiled me to the point that I cant get donuts from anywhere but here. Luckily, it's very close, the employees are always friendly, and I always manage to find street parking despite the constant stream of customers in and out. Nothing compares; you'll know when you try them. I gifted my mother a donut and coffee pairing of the month (which I painstakingly assemble and mail to her in Virginia), and she's always thrilled to get her beckers in the mail. The employees even recently helped me pick my donut to pair, and I lust after the wedding cake in the window every time. All said, I'm now a terribly loyal customer, and I'd imagine my impending-expanding waistline will be a testament
useful	4
funny	3
cool	4

Table 2: Yelp review sample

Our target is analysing each review's "text" field to extract dish and food names.

Also, the dataset includes categories with a one-to-many relationship that indicates the nature and activities of this business. The "Category" table has the following fields:

- id
- business_id
- category

id	43057	43058	43059	43060
business_id	ztmIXLuIAADzuTa4skUTgg			
category	Custom Cakes	Bakeries	Food	Donuts

Table 3: Yelp category sample

As we can see, the same business can fall under more than one category at the same time, some are more generic, and some are very specific.

I used MySQL database engine, and MySQL workbench to handle the database, and explore the data.

3.1 Building the Lookup Dictionary

To build the dataset, I analyse the reviews' text, using Python programming language (Pyhton, 2018), and some libraries like Pandas (Pandas, 2018) and NumPy (NumPy, 2018).

Also, I used NLTK (NLTK, 2018), which is a natural language toolkit, and WordNet (NLTK WordNET, 2018) which is a lexical English database. WordNet has three synsets for food, two of them are relevant to the edible food; which are "food.n.01" and "food.n.02".

I started by making a list of all the nouns in the dictionary that has any relation to these two synsets. NLTK has a *closure* of *hyponyms* list that can get subordinate of a word; the subordinates have "is-a" relationship; *pizza is a food*. Looping through this list, I got 2177 results for "food.n.01", and 1621 result for the "food.n.02". After eliminating repeated and irrelevant results like single letters, the resulting lookup list was 3480 results.

Some items consisted of a single word, e.g. ***bread*** or multiple words e.g. ***English breakfast tea***. Some items consist of another dish name or even more than one e.g. ***banana bread***.

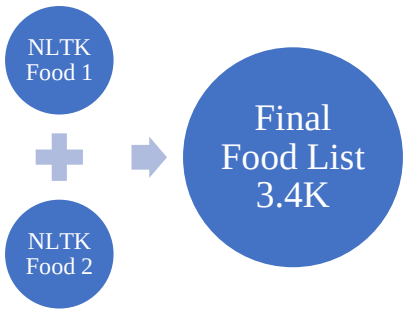

Figure 2: Building food lookup from NLTK dictionary

3.2 Pre-processing the Reviews

Since the reviews are free text, I had to clean the data a little bit, which included removing punctuation, except for the ’ apostrophe symbol, because it will affect the PoS tagging badly; especially in possessive endings. Other punctuation list included !"#$%&()*+,-./:;<=>?@[\]^_`{|}~

Also, the cleaning process included removing new lines within the same review since the line breaker would affect the automated process negatively.

I didn’t exclude the stop words since they will affect the dataset and would result in deforming some dish names as the stop words are part of the name. e.g. ***fish and chips***.

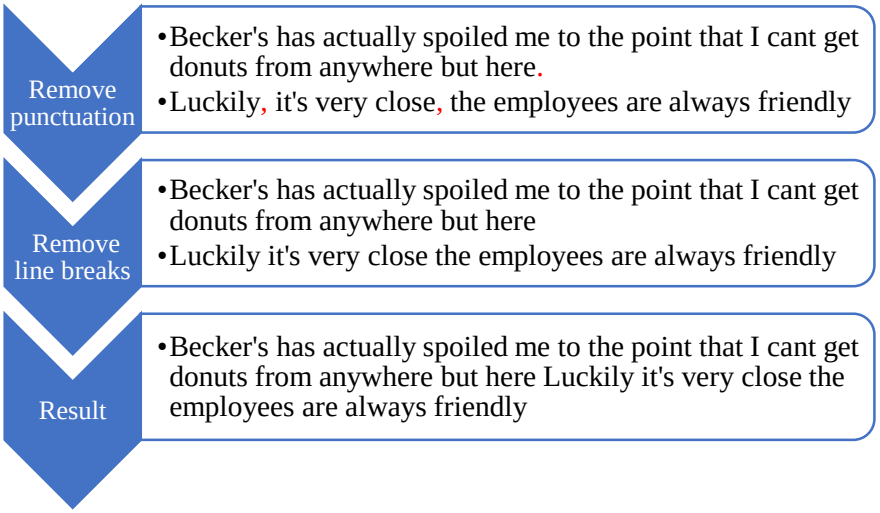

Figure 3: Pre-processing review text

The Yelp dataset includes many categories other than food, so I filtered the food-related reviews only; out of 1294 categories, 61 only of them are related to food; which would refine the dataset a lot.

Yelp food-related categories:

- Food
- Acai Bowls

- Bagels
- Bakeries
- Beer, Wine & Spirits
- Beverage Store
- Breweries
- Brewpubs
- Bubble Tea
- Butcher
- CSA
- Chimney Cakes
- Cideries
- Coffee & Tea
- Coffee Roasteries
- Convenience Stores
- Cupcakes
- Custom Cakes
- Desserts
- Distilleries
- Do-It-Yourself Food
- Donuts
- Empanadas
- Farmers Market
- Food Delivery Services
- Food Trucks
- Gelato
- Grocery
- Honey
- Ice Cream & Frozen Yogurt
- Imported Food
- International Grocery
- Internet Cafes
- Juice Bars & Smoothies
- Kombucha
- Organic Stores
- Patisserie/Cake Shop
- Piadina
- Poke
- Pretzels
- Shaved Ice
- Shaved Snow
- Smokehouse
- Specialty Food
- Candy Stores
- Cheese Shops
- Chocolatiers & Shops
- Fruits & Veggies

- Health Markets
- Herbs & Spices
- Macarons
- Meat Shops
- Olive Oil
- Pasta Shops
- Popcorn Shops
- Seafood Markets
- Street Vendors
- Tea Rooms
- Water Stores
- Wineries
- Wine Tasting Room

To better understand what makes a word a good candidate, I looked at a few sample reviews; I found that our target could be determined by the preceding and following words.

"we ordered the clams"

"I ordered the Spicy Japanese Noodles with chicken"

"I ordered the everything bagel"

"The buffalo pizza was great"

"The beer is great"

"The burgers were great"

We see here that people follow a pattern either with the same words or with the same structure, so it looks promising to consider both the actual word and the Part of Speech (PoS) tags.

3.3 Building the dataset

Looking at the reviews at a document level (for each review), then going one level to sentence level, and even deeper to word level.

So, I started by retrieving 100K reviews, and going through each review tokenizing and tagging each word using NLTK PoS tag (Ahiladas, Saravanaperumal, Balachandran, Sripalan, & Ranathunga, 2015; Harrison, 2018) and considering each noun as a candidate.

NLTK includes the following PoS Tags:

- CC coordinating conjunction
- CD cardinal digit
- DT determiner
- EX existential there (like: “there is” … think of it like “there exists”)
- FW foreign word
- IN preposition/subordinating conjunction
- JJ adjective ‘big’
- JJR adjective, comparative ‘bigger’
- JJS adjective, superlative ‘biggest’
- LS list marker 1)
- MD modal could, will
- NN noun, singular ‘desk’
- NNS noun plural ‘desks’
- NNP proper noun, singular ‘Harrison’
- NNPS proper noun, plural ‘Americans’

- PDT predeterminer ‘all the kids’
- POS possessive ending parent’s
- PRP personal pronoun I, he, she
- PRP$ possessive pronoun my, his, hers
- RB adverb very, silently,
- RBR adverb, comparative better
- RBS adverb, superlative best
- RP particle give up
- TO to go ‘to‘ the store.
- UH interjection errrrrrrrm
- VB verb, base form take
- VBD verb, past tense took
- VBG verb, gerund/present participle taking
- VBN verb, past participle taken
- VBP verb, sing. present, non-3d take
- VBZ verb, 3rd person sing. present takes
- WDT wh-determiner which
- WP wh-pronoun who, what
- WP$ possessive wh-pronoun whose
- WRB wh-abverb where, when

By using the PoS tags that indicates a noun (NN, NNP, NNS, NNPS) I could filter only the nouns (Chinsha & Joseph, 2014), and considered up to 4 preceding and 4 following words, also during this step I considered the PoS tag for the all the aforementioned 9

words (4 before + noun + 4 after), I handled the cases of missing data if the candidate noun doesn't have enough preceding or following words.

I tried considering foreign words (FW), however, they were very misleading, and resulted in many irrelevant words that make no sense, since the PoS tags were confused. And using only nouns resulted in adding some nouns from another language; even when used within an English review, to the candidate list.

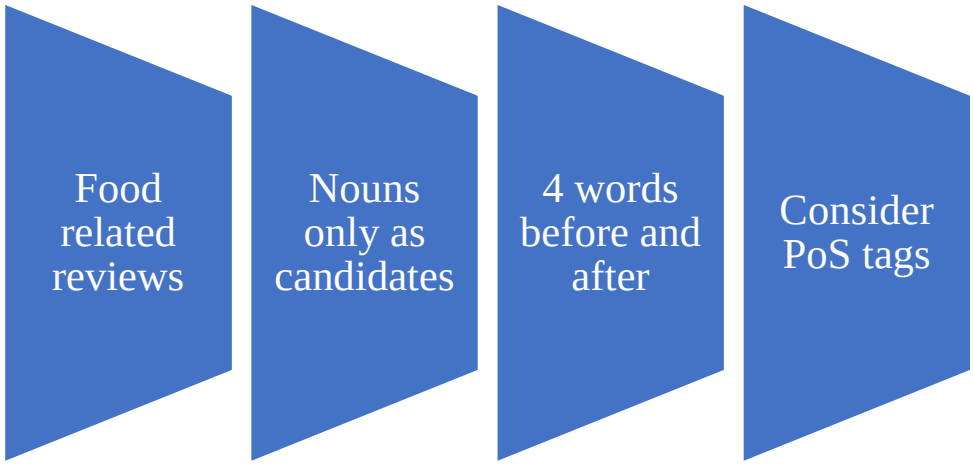

Figure 4: Generating dataset from reviews text

And using the previously generated lookup table of 3.4K words, I labelled the known candidates "Positive" with "1" if they existed in this table, and "Unlabelled" with "0" if they are not in the table. The result was 2.6M result in the following format:

- id
- review_id
- name
- name_type
- before_1
- before_2
- before_3
- before_4
- before_type_1
- before_type_2
- before_type_3
- before_type_4

- after_1
- after_2
- after_3
- after_4
- after_type_1
- after_type_2
- after_type_3
- after_type_4
- is_food

e.g. “gifted my mother a donut and coffee pairing of” the resulting record would be something like this

id		2591971	
review_id		--AmbUiukqrLJpZSSjXO0g	
before_4	gifted	before_type_4	VBD
before_3	my	before_type_3	PRP$
before_2	mother	before_type_2	NN
before_1	a	before_type_1	DT
name	donut	name_type	NN
after_1	and	after_type_1	CC
after_2	coffee	after_type_2	NN
after_3	pairing	after_type_3	NN
after_4	of	after_type_4	IN
is_food		1	

Table 4: Food review sample

The resulting dataset included around 513K positive samples and around 2.1M rows unlabelled entries. The dataset included 54K unique words in total, 1.1K of them (1.91%) were marked as positive using the lookup dictionary.

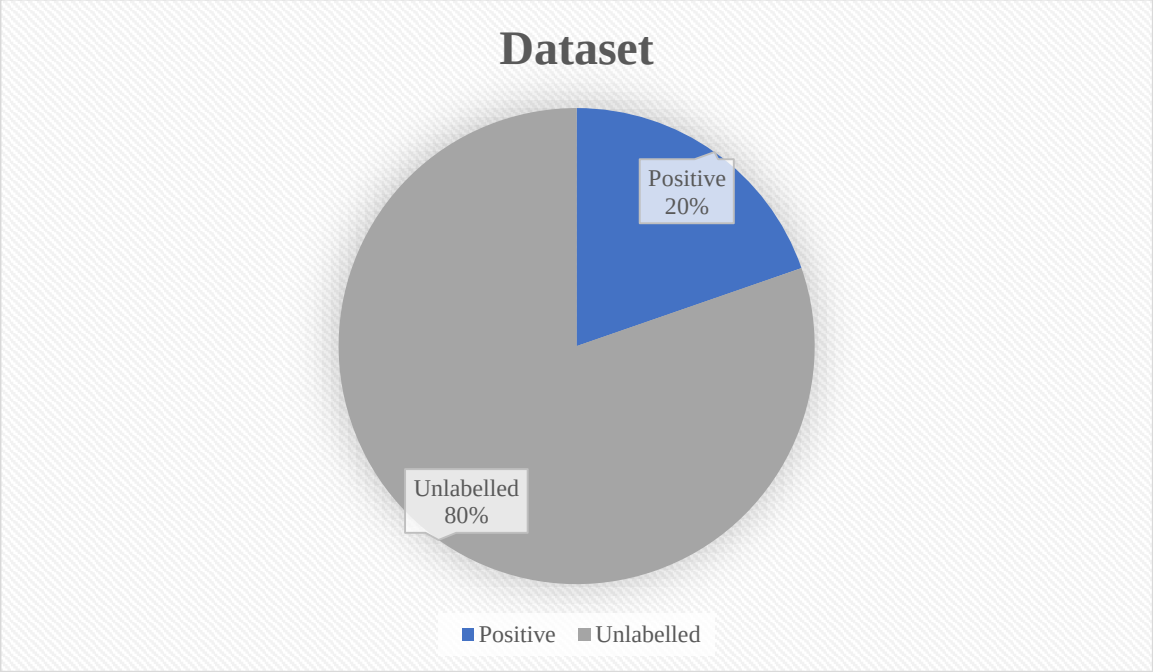

Figure 5: Positive-Unlabeled ratio

Some words were repeated extensively; “place” was used over 54K times and “coffee” was the most frequent positive word with almost 21K times. Around 28K words appeared only once. The table below shows the top 20 most used nouns.

Noun	Count	Noun	Count
place	54733	menu	13237
food	50098	people	12401
service	31142	restaurant	11946
time	29521	chicken	10919
coffee	20971	way	10861
order	17062	day	10319
cream	15157	location	9788
staff	14861	chocolate	9718
tea	13517	everything	9626
ice	13329	store	9360

Table 5: Most common nouns in the dataset

The graph below shows the word represented by an index, and how many times it was repeated.

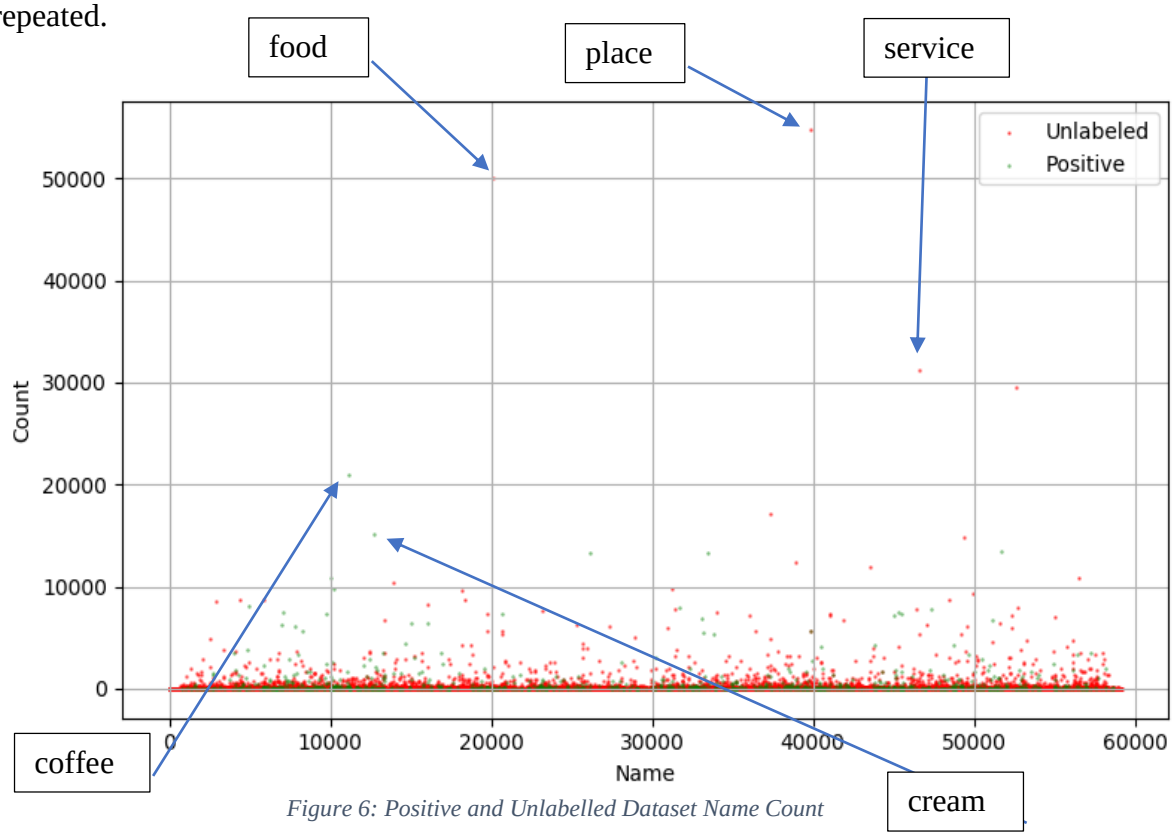

Figure 6: Positive and Unlabelled Dataset Name Count

3.4 Building the model

Since the new dataset includes "Positive" and "Unlabelled" samples, with no negative samples; I use Positive Unlabelled (PU) learning techniques. Transductive PU learning (Mordelet & Vert, 2014) is focused on finding positive samples from the dataset mainly and doesn't care about negative ones, while inductive is focusing on finding whether the candidate is positive or negative. In the case of NER, the main goal is to detect only positive samples, so I use transductive PU learning.

Using python and scikit-learn (Scikit-learn, 2018), which is a library for python that offers machine learning capabilities; like classification, clustering and regression, sometimes it referred to as "sklearn".

The dataset is mainly text, and machine learning modelling can't handle these data directly, I start by pre-processing the data with a label encoding, I used sklearn LabelEncoder which encodes labels with a value between 0 and n_classes-1. This label encoder was applied to the dataset vertically for each column independently. The table below shows a partial sample of the dataset before and after applying the label encoder.

name_type	before_1	after_4	before_type_1	after_type_4	is_food
NN	null	night	NULL	NN	1
NN	my	great	PRP$	JJ	0
NNS	good	they	JJ	PRP	0
NN	great	anything	JJ	NN	1
NN	do	a	VB	DT	0
NN	great	null	JJ	NULL	0
NN	null	and	NULL	CC	0
NN	the	a	DT	DT	0

Table 6: Dataset sample before label encoding

name_type	before_1	after_4	before_type_1	after_type_4	is_food
0	4996	5297	15	12	1
0	4817	3549	19	8	0
3	3160	7839	7	19	0
0	3213	456	7	12	1
0	2169	167	26	4	0
0	3213	5366	7	16	0
0	4996	407	15	2	0
0	7233	167	3	4	0

Table 7: Dataset sample after label encoding

I tried to build the model with a sample size of 500K; the selected sample included 97.7K (19.54%) positive samples with 910 unique ones, and the remaining 402.3K was unlabelled.

The idea of bagging is to have the full positive records coupled with a random sample of the unlabelled ones (include in the bag) with the same size of positive set, and train the model, using Decision Tree, while considering the unlabelled sample included in the bag as negative (W. S. Lee & Liu, 2003; Mordelet & Vert, 2014), then score the model on

the rest of the unlabelled records that weren't chosen for training (out of bag). By repeating this process many times and adding the score of each iteration, we get the probability of each candidate individually. The suggested number of iterations by (Heise, 2017) was 1/6th of the sample size with is around 83K iterations in our case when testing a sample size of 500K.

The PU bagging algorithm gives us a score (0 to 1) of each unlabelled noun with the possibility of being positive. I sorted the results in a descending order to better analyse the results, since I am more interested in the positive ones with higher scores.

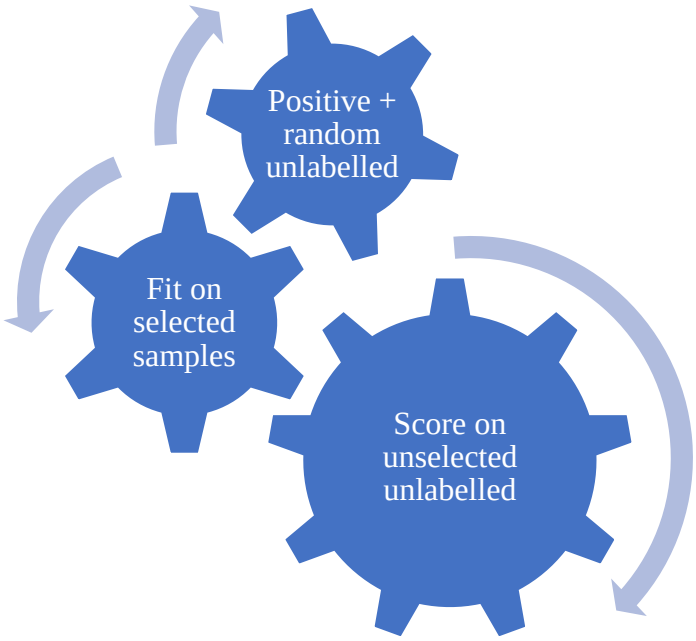

Figure 4: PU using bagging algorithm

3.5 Post-processing

Since the dataset was collected from multiple reviews, I encountered some repeated words in the results, for example, the unlabelled word "gelato" was repeated for 2606 times. The repeated words got multiple different scores from each repetition within the dataset; this is completely different from the scores we get from a repeated scoring

iteration. So, I took the average score of each word and concatenated the repeated results into a single score which narrowed down the 402.3K unlabelled samples down to around 25.7K unique samples.

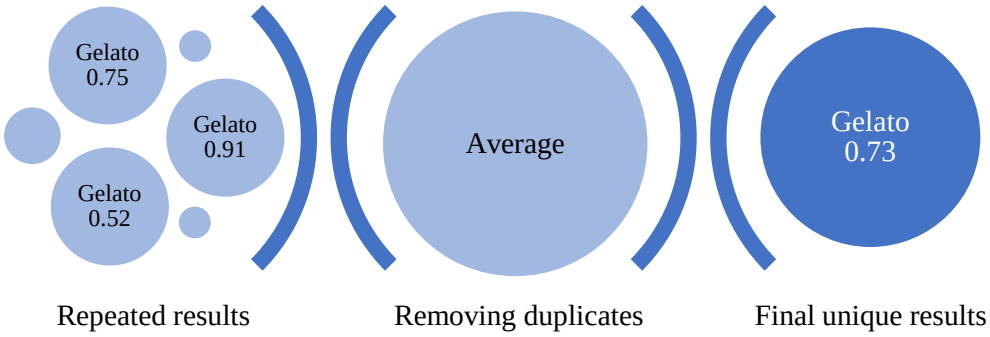

Figure 5: Averaging duplicated results scores

Chapter 4: Results

In this chapter, I discuss the results of existing systems briefly, then I explore the research results in detail.

4.1 Testing Existing Systems

Many systems have been developed with NER in mind, they can achieve a certain level of analysis, yet not that extensive. Testing multiple APIs and tools for Named Entity Recognition, here will briefly review them with a focus on dishes and food.

Amazon comprehend (Amazon, 2018): Entity recognition; yet limited to "commercial item, date, event, location, organization, other, person, quantity, title" only with no option for food or dish. Sentiment analysis offers Neutral, Positive, Negative, and Mixed with confidence for each category, but for the overall text only.

Stanford CoreNLP (Stanford, 2018): their named entity recognition doesn't recognize food at all. But the sentiment is relatively smart since it breaks the text into sentences and gives it a rating from very negative to very positive. It has basic dependencies which are very smart at finding links and relationships.

Google cloud natural language (Google, 2018): Named entity could recognize every food item and classify it as a "consumer good" however there is no sub-class for food only. Categories: it can help classify in which category the whole document can fall under and whether it is food related or not, but no classification for entities other than the aforementioned "consumer good". Sentiment: it's very accurate with a score and magnitude for each sentence. Syntax: advanced analytics and complex relationships.

IBM Watson natural language understanding (IBM, 2018): Named entity: it can barely recognize any entities; mainly persons and locations. The sentiment is relatively

accurate Categories: it can recognize the whole text category: like “food and drink” similar to Google’s.

4.2 Research Results

As mentioned before the results here are the probability of each unlabelled noun of being positive. To evaluate NER systems, I perform a comparison between the generated labels from the system and the labels generated by humans; preferably linguists for the same candidate (Nadeau & Sekine, 2007). Entity recognition patterns are measured by how they can extract more true positive results while keeping the false positive ratio low (Gupta & Manning, 2014). Here we sort the unlabelled dataset by probability in descending order and we check the top 500 highest ranked results.

Looking at the results, I had to review them manually (Ahiladas et al., 2015), it falls under one category out of 6; if it is Correct (1), Typo (2), Restaurant name (3), Part (4), Description (5), irrelevant (6) and Wrong (0). Here we have a sample of each category:

Correct (1): if the candidate name is for sure a food or dish name

e.g., “The food is wonderful. I always end up getting a *chimichanga*. And churros. Fresh, hot churros.”

Chimichanga is a deep-fried burrito

Typo (2): if the unlabelled name is a dish or food name with a spelling mistake

“I order a *tukery* sandwhich, with mayo.”

Misspelt *turkey*

Restaurant (3): if the reviewer is referring to the restaurant name as a food

“Satisfies my craving for *wienerschnitzle* and other delectable treats.”

Wienerschnitzel is an American hot dog’s fast food chain

Part (4): if the candidate name is a part of a dish name; some users prefer using only part of the dish name in the review (Chao et al., 2016)

“and needed to take out a few things in the end i ordered the pad *tai*.”.

The full dish name is Pad *Tai*

Description (5): if the review is describing the food texture, presentation, features, etc.

“If your going for *gormet* or great steaks”

Gormet is describing the steak

Irrelevant (6): if the review is in another language, or under wrong business

“a giant rash appeared on my face, I needed some *cortisone* cream asap”

Cortisone is referring to a medical item and reviewing non-food business

Wrong (0): “The in-house *mixologist* at this restaurant truly knows what he's doing.”

Mixologist is referring to a person

The result was as indicated below

Category		Count	Percentage	Total
1	Correct	248	50%	67 %
2	Typo	50	10%	
3	Restaurant	4	1%	
4	Part	35	7%	
5	Description	50	10%	33%
0	Wrong	81	16%	
6	Irrelevant	32	6%	

Table 8: Top 500 results

I managed, using the system, to discover some Italian, Chinese and Japanese and many other non-English dishes, also it discovered some local names for normal dishes e.g. snag for Australian sausage.

I set a cut-off of 0.8 that's almost within the top 500 ranking results, and I use the remaining 2.1M samples that were not included in building the model to test whether how many of them are recognized as a dish using the results from NLTK lookup dictionary versus the framework results (lookup + the model with a 0.8 cut-off) versus lookup and manually checked results, and then compare the three results.

After eliminating the repeated words from the 2.1M samples I have 52.8K unique results.

Total	52817		
Method	Lookup	Lookup + Model 0.8 cutoff	Lookup + Manual Verification
Positive Count	1101	1369	1275
Positive Ratio	2.08%	2.59%	2.41%
Newly discovered		24.34%	15.80%

Table 9: Results comparison: lookup, model, and manual for test data

Looking at the results we see that using the lookup table built from dictionary managed to discover 1101 dish names, when adding the model results we could achieve around 24.34% newly discovered entities, based on the accuracy from table 7 I expect that only 67% will be correct, using manual labelling results showed 15.80 %, which is approximately the result of 67% of 24%.

The graph below shows the number of recognized entities for each method.

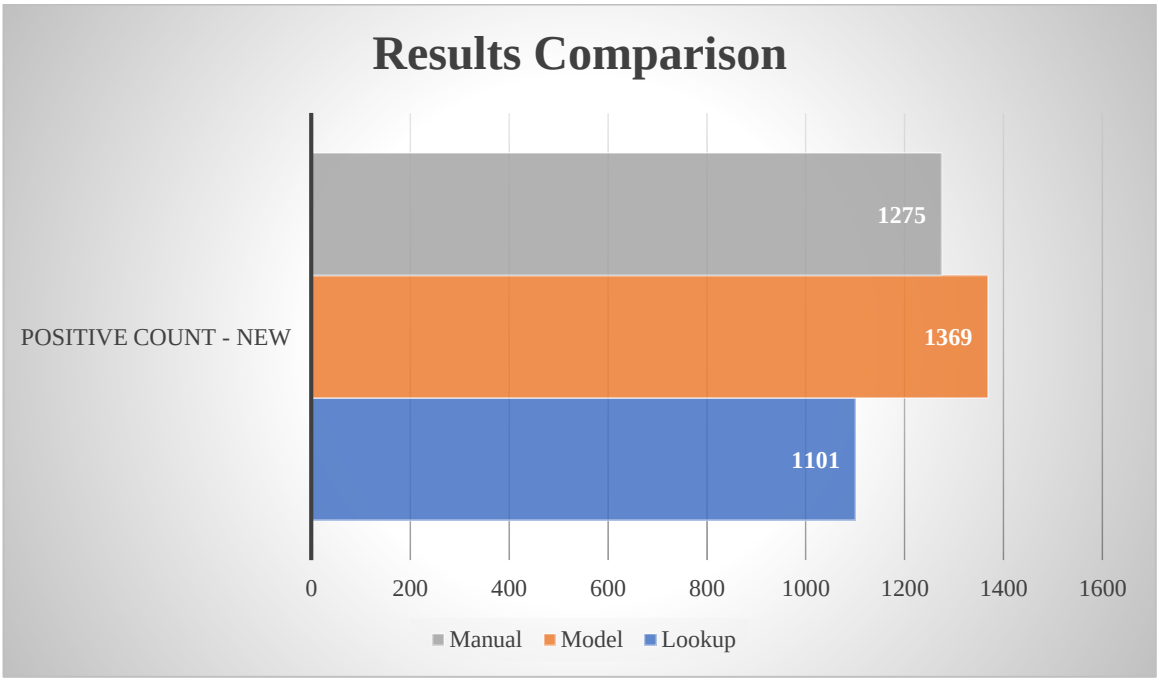

Figure 6: Positive count results

4.3 Discussion

I introduced a framework for automating named entity recognition, starting by generating the initial lookup table of over 3.4K items. Followed by building a dataset, considering 4 words preceding and following the candidate word, and including PoS tags as well, then labelling all the nouns found in the reviews' text.

The framework is algorithm independent, yet for testing purposes, I used Decision Tree, it can be extended to many other algorithms even using neural networks or deep learning (Skabar, 2002). Since the model was built and scored for each item in the dataset individually, considering repetitions in the dataset, I calculate the average and eliminate duplicates. From practice, the accuracy can be improved by increasing the sample size used in modelling, also by trying to filter out non-English reviews.

The research was focused on 1-gram NLP, however, the system was built with expansion in mind, to include dish names that consist of multiple words.

NER in other languages depends on the availability and capability of natural language processing libraries. Libraries like Stanford CoreNLP support some of the features lists (Please refer to appendix Stanford CoreNLP language supported features)

The use of the results of the research can be used as a seed of labelled data to the next iteration, which would result in more recognised entities over time (Gupta & Manning, 2014).

Chapter 5: Conclusion

To sum it all, the system is using NLP techniques to recognize dish and food names as a named entity. At the beginning of the research we had four main questions to answer; concerning the feasibility of analysing free text reviews, overcoming the cold start, which features would help in NER, and the possibility of using PU learning techniques in solving NER challenges.

To overcome the cold start problem and the lack of labelled dataset and avoid manually labelling the dataset; which seems impossible considering the research scope and time, I used NLTK dictionary to build a lookup table of food-related items. Using this lookup table as a source to discover some positive samples within the dataset, which lead to having around 513K positive samples automatically.

To tackle the NER challenges, I looked into some review samples, to find what makes a word a good candidate as a dish or food name. I discovered that the surrounding words of a possible candidate noun can indicate whether this noun is a food or dish name or not; so, I choose the four preceding words and the four following ones, also I included the PoS tags which explain a lot how a sentence is structured because in free text people tend to use different vocabulary based on the education, background, feeling, etc... which has helped in building the model.

Using PU learning techniques solved the challenge of having only positive samples and no negative samples. I implemented the bagging technique which takes many iterations yet achieves a good accuracy.

The system managed to recognize dishes and food names from free-text reviews with an accuracy of 67%. And PU learning contributed to about 15% improvement over using a lookup dictionary.

This framework has a potential for expansion to be used with more than 1 word, also can be extended to cover many other fields, and its algorithm independent.

References

Ahiladas, B., Saravanaperumal, P., Balachandran, S., Sripalan, T., & Ranathunga, S. (2015). Ruchi: Rating individual food items in restaurant reviews. In *Proceedings of the 12th International Conference on Natural Language Processing* (pp. 209–214).

Alfonseca, E., & Manandhar, S. (2002). An unsupervised method for general named entity recognition and automated concept discovery. In *Proceedings of the 1st international conference on general WordNet, Mysore, India* (pp. 34–43).

Amazon. (2018). Amazon Comprehend. Retrieved May 30, 2018, from https://console.aws.amazon.com/comprehend/home

Bloomberg. (2019). Bloomberg. Retrieved from https://www.bloomberg.com/

Bontcheva, K., Derczynski, L., & Roberts, I. (2017). Crowdsourcing named entity recognition and entity linking corpora. In *Handbook of Linguistic Annotation* (pp. 875–892). Springer.

Bowden, K. K., Wu, J., Oraby, S., Misra, A., & Walker, M. (2018). SlugNERDS: A Named Entity Recognition Tool for Open Domain Dialogue Systems. *ArXiv Preprint ArXiv:1805.03784*.

Brin, S. (1998). Extracting patterns and relations from the world wide web. In *International Workshop on The World Wide Web and Databases* (pp. 172–183). Springer.

Chao, C.-Y., Chu, Y.-F., Ho, Y., Wang, C.-J., & Tsai, M.-F. (2016). Dish Discovery via Word Embeddings on Restaurant Reviews. In *RecSys Posters*.

Chinsha, T. C., & Joseph, S. (2014). Aspect based opinion mining from restaurant reviews. *International Journal of Computer Applications*, *975*, 8887.

Claesen, M., De Smet, F., Suykens, J. A. K., & De Moor, B. (2015). A robust ensemble approach to learn from positive and unlabeled data using SVM base models. *Neurocomputing*, *160*, 73–84.

Collins, M., & Singer, Y. (1999). Unsupervised models for named entity classification. In *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

CoreNLP, S. (2018). Using Stanford CoreNLP on other human languages. Retrieved from https://stanfordnlp.github.io/CoreNLP/human-languages.html

Dale, R. (2018). Text analytics APIs, Part 1: The bigger players. *Natural Language Engineering*, *24*(2), 317–324.

Elkan, C., & Noto, K. (2008). Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 213–220). ACM.

Fan, M., & Khademi, M. (2014). Predicting a business star in yelp from its reviews text

alone. *ArXiv Preprint ArXiv:1401.0864*.

Google. (2018). Google cloud natural language. Retrieved May 30, 2018, from https://cloud.google.com/natural-language/

Gupta, S., & Manning, C. (2014). Improved pattern learning for bootstrapped entity extraction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning* (pp. 98–108).

Harrison. (2018). NLTK Part of Speech Tagging. Retrieved from https://pythonprogramming.net/natural-language-toolkit-nltk-part-speech-tagging/

Heise, P. H. (2017). PU Bagging. Retrieved from https://github.com/phuijse/bagging_pu

Huang, J., Rogers, S., & Joo, E. (2014). Improving restaurants by extracting subtopics from yelp reviews. *IConference 2014 (Social Media Expo)*.

IBM. (2018). IBM Watson natural language understanding. Retrieved May 30, 2018, from https://www.ibm.com/watson/services/natural-language-understanding/

Ji, H., & Grishman, R. (2006). Data selection in semi-supervised learning for name tagging. In *Proceedings of the Workshop on Information Extraction Beyond The Document* (pp. 48–55). Association for Computational Linguistics.

Kaboutari, A., Bagherzadeh, J., & Kheradmand, F. (2014). An evaluation of two-step techniques for positive-unlabeled learning in text classification. *Int. J. Comput. Appl. Technol. Res*, *3*, 592–594.

Kim, W. G., Li, J. J., & Brymer, R. A. (2016). The impact of social media reviews on restaurant performance: The moderating role of excellence certificate. *International Journal of Hospitality Management*, *55*, 41–51.

Lee, S., & Ro, H. (2016). The impact of online reviews on attitude changes: the differential effects of review attributes and consumer knowledge. *International Journal of Hospitality Management*, *56*, 1–9.

Lee, W. S., & Liu, B. (2003). Learning with positive and unlabeled examples using weighted logistic regression. In *ICML* (Vol. 3, pp. 448–455).

Li, C., & Hua, X.-L. (2014). Towards positive unlabeled learning for parallel data mining: a random forest framework. In *International Conference on Advanced Data Mining and Applications* (pp. 573–587). Springer.

Li, X.-L., & Liu, B. (2005). Learning from positive and unlabeled examples with different data distributions. In *European Conference on Machine Learning* (pp. 218–229). Springer.

Lin, W., Yangarber, R., & Grishman, R. (2003). Bootstrapped learning of semantic classes from positive and negative examples. In *Proceedings of ICML-2003 Workshop on The Continuum from Labeled to Unlabeled Data* (Vol. 1, p. 21).

Mordelet, F., & Vert, J.-P. (2014). A bagging SVM to learn from positive and unlabeled examples. *Pattern Recognition Letters*, *37*, 201–209.

Morris, H., & Edalat, N. (2015). *Impact of Social Media in the Restaurant Industry*. *Emirates Academy of Hospitality Management* (Vol. 1).

Nadeau, D., & Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, *30*(1), 3–26.

Nadeau, D., Turney, P. D., & Matwin, S. (2006). Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In *Conference of the Canadian Society for Computational Studies of Intelligence* (pp. 266–277). Springer.

NLTK. (2018). NLTK. Retrieved from http://www.nltk.org/

NLTK WordNET. (2018). NLTK WordNet. Retrieved May 30, 2018, from http://www.nltk.org/howto/wordnet.html

NumPy. (2018). NumPy. Retrieved from http://www.numpy.org/

Oman, E. (2016). Where can I find a text list or library that contains a list of common foods? Retrieved from https://stackoverflow.com/posts/20842943/revisions

Pandas. (2018). Pandas. Retrieved from https://pandas.pydata.org/

Pyhton. (2018). Python. Retrieved from https://www.python.org/

Ren, X., El-Kishky, A., Wang, C., & Han, J. (2016). Automatic entity recognition and typing in massive text corpora. In *Proceedings of the 25th International Conference Companion on World Wide Web* (pp. 1025–1028). International World Wide Web Conferences Steering Committee.

Rossetti, M., Stella, F., & Zanker, M. (2016). Analyzing user reviews in tourism with topic models. *Information Technology & Tourism*, *16*(1), 5–21.

Sansone, E., De Natale, F. G. B., & Zhou, Z.-H. (2018). Efficient training for positive unlabeled learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Scikit-learn. (2018). Scikit-learn. Retrieved from https://scikit-learn.org/

Skabar, A. (2002). Single-class classifier learning using neural networks: extracting context from unlabeled data. *Artificial Intelligence and Applications (AIA2002), Malaga, Spain*.

Stanford. (2018). Stanford CoreNLP. Retrieved May 30, 2018, from http://corenlp.run/%5Cnhttp://nlp.stanford.edu/software/corenlp.shtml

TripAdvisor. (2019). TripAdvisor. Retrieved from https://www.tripadvisor.com/

Wright, R. (2017). Positive-unlabeled learning. Retrieved from https://roywright.me/2017/11/16/positive-unlabeled-learning/

Yelp. (2019). Yelp. Retrieved May 30, 2018, from https://www.yelp.com/

Zhang, Z., Ye, Q., Law, R., & Li, Y. (2010). The impact of e-word-of-mouth on the online popularity of restaurants: A comparison of consumer reviews and editor reviews. *International Journal of Hospitality Management*, *29*(4), 694–700.

Zhao, S., Han, S., Meng, R., He, D., & Zhang, D. (2017). Learning Semantic Representation from Restaurant Reviews: A Study of Yelp Dataset. *IConference 2017 Proceedings Vol. 2*.

Appendices

Code Samples

FillData.py

Part of the code is adopted from online code snippet (Oman, 2016)

```
from nltk.corpus import wordnet as wn
import mysql.connector
import os
import pandas as pd
import string
import nltk
from nltk.stem.porter import PorterStemmer
from nltk.corpus import stopwords
from _mysql import NULL
import json

try:
    cnx = mysql.connector.connect(user='USERNAME', password='PASSWORD',
host='SERVER_IP', database='yelp_db')
    cursor = cnx.cursor()

    cursor.execute("""select distinct review.id, review.text from yelp_db.review inner
join yelp_db.category using(business_id) where
                    category = 'Food' or
                    category = 'Acai Bowls' or
                    category = 'Bagels' or
                    category = 'Bakeries' or
                    category = 'Beer, Wine & Spirits' or
                    category = 'Beverage Store' or
                    category = 'Breweries' or
                    category = 'Brewpubs' or
                    category = 'Bubble Tea' or
                    category = 'Butcher' or
                    category = 'CSA' or
                    category = 'Chimney Cakes' or
                    category = 'Cideries' or
                    category = 'Coffee & Tea' or
                    category = 'Coffee Roasteries' or
                    category = 'Convenience Stores' or
                    category = 'Cupcakes' or
                    category = 'Custom Cakes' or
                    category = 'Desserts' or
                    category = 'Distilleries' or
```

```
                    category = 'Do-It-Yourself Food' or
                    category = 'Donuts' or
                    category = 'Empanadas' or
                    category = 'Farmers Market' or
                    category = 'Food Delivery Services' or
                    category = 'Food Trucks' or
                    category = 'Gelato' or
                    category = 'Grocery' or
                    category = 'Honey' or
                    category = 'Ice Cream & Frozen Yogurt' or
                    category = 'Imported Food' or
                    category = 'International Grocery' or
                    category = 'Internet Cafes' or
                    category = 'Juice Bars & Smoothies' or
                    category = 'Kombucha' or
                    category = 'Organic Stores' or
                    category = 'Patisserie/Cake Shop' or
                    category = 'Piadina' or
                    category = 'Poke' or
                    category = 'Pretzels' or
                    category = 'Shaved Ice' or
                    category = 'Shaved Snow' or
                    category = 'Smokehouse' or
                    category = 'Specialty Food' or
                    category = 'Candy Stores' or
                    category = 'Cheese Shops' or
                    category = 'Chocolatiers & Shops' or
                    category = 'Fruits & Veggies' or
                    category = 'Health Markets' or
                    category = 'Herbs & Spices' or
                    category = 'Macarons' or
                    category = 'Meat Shops' or
                    category = 'Olive Oil' or
                    category = 'Pasta Shops' or
                    category = 'Popcorn Shops' or
                    category = 'Seafood Markets' or
                    category = 'Street Vendors' or
                    category = 'Tea Rooms' or
                    category = 'Water Stores' or
                    category = 'Wineries' or
                    category = 'Wine Tasting Room'
                    limit 100000;""")

    results = cursor.fetchall()
except mysql.connector.Error as err:
  if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
    print("Something is wrong with your user name or password")
  elif err.errno == errorcode.ER_BAD_DB_ERROR:
```

```
    print("Database does not exist")
  else:
    print(err)
else:
#     print("closed")
    cnx.close()

reviews = list(zip(*results))
reviews_id = reviews[0]
reviews = reviews[1]

sp = string.punctuation
sp = sp.replace('\', '')

reviews = list(map(lambda t: ''.join([' ' if c in sp else c for c in t]), reviews))
reviews = list(map(lambda t: ''.join([' ' if c == '\n' else c for c in t]), reviews))

food_1 = wn.synset('food.n.01')
food_2 = wn.synset('food.n.02')

set_food_1 = set([w for s in food_1.closure(lambda s:s.hyponyms()) for w in
s.lemma_names()])
set_food_2 = set([w for s in food_2.closure(lambda s:s.hyponyms()) for w in
s.lemma_names()])

all_foods = list(set_food_1) + list(set_food_2 - set_food_1)
all_foods = [item.lower() for item in all_foods]
all_foods = list(map(lambda t: ''.join([' ' if c in sp else c for c in t]), all_foods))
all_foods.sort()
f = open("all_foods.txt", "w")
f.writelines(["%s\n" % item.lower() for item in all_foods if len(item) > 1])

try:
    cnx = mysql.connector.connect(user='root', password='123456', host='127.0.0.1',
database='review_data')
    cursor = cnx.cursor()
except mysql.connector.Error as err:
  if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
    print("Something is wrong with your user name or password")
  elif err.errno == errorcode.ER_BAD_DB_ERROR:
    print("Database does not exist")
  else:
    print(err)

    cnx.close()

results_list = []
```

```
i = 1
for review in reviews:
    print(i, " of ", len(reviews))
    review_id = reviews_id[i-1]
    i += 1
    tokens = nltk.word_tokenize(review)
    tagged = nltk.pos_tag(tokens)

    for (index, (word, pos) ) in enumerate(tagged):
        if ((pos == 'NN' or pos == 'NNP' or pos == 'NNS' or pos == 'NNPS') and
len(word) > 2):
            if(index > 4):
                before_4 = tagged[index - 4][0]
                before_3 = tagged[index - 3][0]
                before_2 = tagged[index - 2][0]
                before_1 = tagged[index - 1][0]
                before_type_4 = tagged[index - 4][1]
                before_type_3 = tagged[index - 3][1]
                before_type_2 = tagged[index - 2][1]
                before_type_1 = tagged[index - 1][1]
            else:
                before_4 = NULL;
                before_type_4 = NULL;
                if(index > 3):
                    before_3 = tagged[index - 3][0]
                    before_2 = tagged[index - 2][0]
                    before_1 = tagged[index - 1][0]
                    before_type_3 = tagged[index - 3][1]
                    before_type_2 = tagged[index - 2][1]
                    before_type_1 = tagged[index - 1][1]
                else:
                    before_3 = NULL;
                    before_type_3 = NULL;
                    if(index > 2):
                        before_2 = tagged[index - 2][0]
                        before_1 = tagged[index - 1][0]
                        before_type_2 = tagged[index - 2][1]
                        before_type_1 = tagged[index - 1][1]
                    else:
                        before_2 = NULL;
                        before_type_2 = NULL;
                        if(index > 1):
                            before_1 = tagged[index - 1][0]
                            before_type_1 = tagged[index - 1][1]
                        else:
                            before_1 = NULL;
                            before_type_1 = NULL;
```

```
        name = word
        is_food = int(name.lower() in all_foods)
        if(index < len(tagged) - 4):
            after_4 = tagged[index + 4][0]
            after_3 = tagged[index + 3][0]
            after_2 = tagged[index + 2][0]
            after_1 = tagged[index + 1][0]
            after_type_4 = tagged[index + 4][1]
            after_type_3 = tagged[index + 3][1]
            after_type_2 = tagged[index + 2][1]
            after_type_1 = tagged[index + 1][1]
        else:
            after_4 = NULL;
            after_type_4 = NULL;
            if(index < len(tagged) - 3):
                after_3 = tagged[index + 3][0]
                after_2 = tagged[index + 2][0]
                after_1 = tagged[index + 1][0]
                after_type_3 = tagged[index + 3][1]
                after_type_2 = tagged[index + 2][1]
                after_type_1 = tagged[index + 1][1]
            else:
                after_3 = NULL;
                after_type_3 = NULL;
                if(index < len(tagged) - 2):
                    after_2 = tagged[index + 2][0]
                    after_1 = tagged[index + 1][0]
                    after_type_2 = tagged[index + 2][1]
                    after_type_1 = tagged[index + 1][1]
                else:
                    after_2 = NULL;
                    after_type_2 = NULL;
                    if(index < len(tagged) - 1):
                        after_1 = tagged[index + 1][0]
                        after_type_1 = tagged[index + 1][1]
                    else:
                        after_1 = NULL;
                        after_type_1 = NULL;

        results_list = [review_id, name.lower(), pos,
                        before_4.lower(), before_3.lower(), before_2.lower(),
before_1.lower(),
                        after_1.lower(), after_2.lower(), after_3.lower(), after_4.lower(),
                        before_type_4, before_type_3, before_type_2, before_type_1,
                        after_type_1, after_type_2, after_type_3, after_type_4,
                        is_food]
        cursor.execute("insert into review_data.food_formatted (review_id, name,
name_type,"
```

```
            "before_4, before_3, before_2, before_1,"
            "after_1, after_2, after_3, after_4,"
            "before_type_4, before_type_3, before_type_2, before_type_1,"
            "after_type_1, after_type_2, after_type_3, after_type_4,"
            "is_food) VALUES(%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,
%s, %s, %s, %s, %s, %s, %s, %s)",
            results_list)

cnx.commit()
cnx.close()
print("done")
```

DishPuLearning.py

Part of the code is adopted from online code snippet (Heise, 2017) and from (Mordelet

& Vert, 2014)

```
import pandas as pd
import numpy as np
import mysql.connector
import csv
import time

from decimal import *
from sklearn.datasets import make_moons
from sklearn.tree import DecisionTreeClassifier
from sklearn import preprocessing
from _mysql import result
from sklearn.preprocessing import LabelBinarizer, LabelEncoder
from collections import defaultdict
from numpy import shape
import pickle
import graphviz
from sklearn import tree

try:
    cnx = mysql.connector.connect(user='root', password='123456', host='127.0.0.1',
database='review_data')
    cursor = cnx.cursor()
except mysql.connector.Error as err:
  if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
    print("Something is wrong with your user name or password")
  elif err.errno == errorcode.ER_BAD_DB_ERROR:
    print("Database does not exist")
  else:
    print(err)
    cnx.close()

cursor.execute("desc review_data.food_formatted;")
column_names =  cursor.fetchall()
column_names = list(zip(*column_names))
column_names = column_names[0]

cursor.execute("select * from review_data.food_formatted limit 500000;")
results = cursor.fetchall()
cnx.close()

N = len(results)
```

```
print(N)

df = pd.DataFrame(results, columns = column_names)
food_list = df['name'].values
df = df.drop(['id', 'review_id', 'name'], axis=1)
str_cols = df.columns[:]
clfs = {c:LabelEncoder() for c in str_cols}
for col, clf in clfs.items():
    df[col] = clfs[col].fit_transform(df[col])

y = df['is_food'].values
X = df.drop(['is_food'], axis=1).values

data_P = X[y==1]
data_U = X[y==0]

NP = data_P.shape[0]
NU = data_U.shape[0]
print("Positive ", NP)
print("Unlabeled ", NU)

T = int(N/6)
K = NP
train_label = np.zeros(shape=(NP+K,))
train_label[:NP] = 1.0
n_oob = np.zeros(shape=(NU,))
f_oob = np.zeros(shape=(NU, 2))

try:
    a = Decimal(time.time())
    for i in range(T):
        print(i, " of ", T)
        bootstrap_sample = np.random.choice(np.arange(NU), replace=True, size=K)
        data_bootstrap = np.concatenate((data_P, data_U[bootstrap_sample, :]), axis=0)

        model = LogisticRegression(C = 1e5, class_weight='balanced', solver='liblinear')
        model.fit(data_bootstrap, train_label)
        idx_oob = sorted(set(range(NU)) - set(np.unique(bootstrap_sample)))
        f_oob[idx_oob] += model.predict_proba(data_U[idx_oob])
        n_oob[idx_oob] += 1
    print("Execution time ", Decimal(time.time()) - a)
    predict_proba = f_oob[:, 1]/n_oob
except Exception as ex:
    print(ex)

# averages
candidates = food_list[y==0]
scores = defaultdict(float)
```

```
count = defaultdict(int)

for candid, result in zip(candidates, predict_proba):
    count[candid] += 1
    scores[candid] += (result - scores[candid]) / count[candid]

predicted_U_avg = scores.items()
predicted_U_count = count.items()

predicted_U_avg = sorted(predicted_U_avg, key=lambda predicted_U_avg:
predicted_U_avg[1], reverse=True)
predicted_U_count = sorted(predicted_U_count, key=lambda predicted_U_count:
predicted_U_count[1], reverse=True)

f = open("predicted_avg.csv", "w", newline='\n', encoding="utf-8")
wr = csv.writer(f)
wr.writerows(predicted_U_avg)
f.close()

f = open("predicted_count.csv", "w", newline='\n', encoding="utf-8")
wr = csv.writer(f)
wr.writerows(predicted_U_count)
f.close()
print("done")
```

Stanford CoreNLP language supported features

This list is available at (CoreNLP, 2018)

Annotator	**ar**	**zh**	**en**	**fr**	**de**	**es**
Tokenize / Segment	✔	✔	✔	✔		✔
Sentence Split	✔	✔	✔	✔	✔	✔
Part of Speech	✔	✔	✔	✔	✔	✔
Lemma			✔			
Named Entities		✔	✔		✔	✔
Constituency Parsing	✔	✔	✔	✔	✔	✔
Dependency Parsing		✔	✔	✔	✔	
Sentiment Analysis			✔			
Mention Detection		✔	✔			
Coreference		✔	✔			
Open IE			✔			