Problem to Solve

In this project, we try to get **what people want to express** in their yelp reviews and turn them into **a form of information** that can be utilized by other machine learning services (prediction, recommendation, etc).

Reviews

Customers put a lot in their reviews. Their feelings, their feedback, their ideas, etc. Everything they want to share with other customers and the services is in the reviews.

Not only we can get **information about the services** in the review, but also **information about the customers** as well. The former let customers know more about the service, the later let services learn more about their customers.

Yelp has reviews for all kinds of services. For clarity, in the rest of the document, the term ‘service’ will be referring to the restaurant service. The term ‘review’ will be referring to the review of the restaurant service.

The Use of Yelp Review

- Help customers decide what to eat by looking into what others like/dislike
- Help customers decide whether to go to the restaurant by looking into other’s feelings and feedbacks about it
- Help restaurants improve their services
- Help restaurants learn the preferences about their customers

Beyond the Human

Even though in recent years there is a huge improvement in Text Analysis technologies, human judgement is still needed for understanding the text messages.

However, machine learning techniques can greatly reduce the cost of it. By prioritizing the reviews or extracting the main topics from the review, it's much easier for customers and service providers to find what they need from the reviews.

This is particularly beneficial for large service providers which may have tons of reviews and feedbacks, making the analysis process more efficient and scalable.

The Scope of This Project

In this project, we narrow the scope of service to the restaurant and the scope of the review to the Toronto area.

Data to Wrangle

Data Source

Yelp dataset is used, the dataset can be directly downloaded from https://www.yelp.com/dataset.

business.json

Contains business data including location data, attributes, and categories.

```
{
    // string, 22 character unique string business id
    "business_id": "tnhfDv5Il8EaGSXZGiuQGg",
    // string, the business's name
    "name": "Garaje",
    // string, the full address of the business
    "address": "475 3rd St",
    // string, the city
```

```
"city": "San Francisco",
// string, 2 character state code, if applicable
"state": "CA",
// string, the postal code
"postal code": "94107",
// float, latitude
"latitude": 37.7817529521,
// float, longitude
"longitude": -122.39612197,
// float, star rating, rounded to half-stars
"stars": 4.5,
// integer, number of reviews
"review_count": 1198,
// integer, 0 or 1 for closed or open, respectively
"is_open": 1,
// object, business attributes to values. note: some attribute values might be objects
"attributes": {
    "RestaurantsTakeOut": true,
    "BusinessParking": {
        "garage": false,
        "street": true,
        "validated": false,
        "lot": false,
        "valet": false
    },
},
// an array of strings of business categories
"categories": [
    "Mexican",
    "Burgers",
    "Gastropubs"
],
    // an object of key day to value hours, hours are using a 24hr clock
    "hours": {
        "Monday": "10:00-21:00",
        "Tuesday": "10:00-21:00",
        "Friday": "10:00-21:00",
        "Wednesday": "10:00-21:00",
        "Thursday": "10:00-21:00",
        "Sunday": "11:00-18:00",
        "Saturday": "10:00-21:00"
    }
}
```

review.json

Contains full review text data including the user_id that wrote the review and the business_id the review is written for.

```
{
    // string, 22 character unique review id
    "review_id": "zdSx_SD6obEhz9VrW9uAWA",
    // string, 22 character unique user id, maps to the user in user.json
    "user_id": "Ha3iJu77CxlrFm-vQRs_8g",
    // string, 22 character business id, maps to business in business.json
    "business_id": "tnhfDv5Il8EaGSXZGiuQGg",
    // integer, star rating
    "stars": 4,
    // string, date formatted YYYY-MM-DD
    "date": "2016-03-09",
    // string, the review itself
    "text": "Great place to hang out after work: the prices are decent, and the ambience is fun. It's a bit loud, but very lively. The staff is friendly, and the food is good. They have a good selection of drinks.",
    // integer, number of useful votes received
    "useful": 0,
    // integer, number of funny votes received
    "funny": 0,
    // integer, number of cool votes received
    "cool": 0
}
```

user.json

User data is not directly used in the project, however we can see taht it includes the user's friend mapping and all the metadata associated with the user which can be used to learn more about the user.

```
{
    // string, 22 character unique user id, maps to the user in user.json
    "user_id": "Ha3iJu77CxlrFm-vQRs_8g",
    // string, the user's first name
    "name": "Sebastien",
    // integer, the number of reviews they've written
    "review_count": 56,
    // string, when the user joined Yelp, formatted like YYYY-MM-DD
    "yelping_since": "2011-01-01",
    // array of strings, an array of the user's friend as user_ids
    "friends": [
        "wqoXYLWmpkEH0YvTmHBsJQ",
        "KUXLLiJGrjtSsapmxmpvTA",
        "6e9rJKQC3n0RSKyHLViL-Q"
    ],
    // integer, number of useful votes sent by the user
    "useful": 21,
    // integer, number of funny votes sent by the user
    "funny": 88,
    // integer, number of cool votes sent by the user
    "cool": 15,
    // integer, number of fans the user has
    "fans": 1032,
    // array of integers, the years the user was elite
    "elite": [
        2012,
        2013
    ],
    // float, average rating of all reviews
    "average_stars": 4.31,
    // integer, number of hot compliments received by the user
    "compliment_hot": 339,
    // integer, number of more compliments received by the user
    "compliment_more": 668,
    // integer, number of profile compliments received by the user
    "compliment_profile": 42,
    // integer, number of cute compliments received by the user
    "compliment_cute": 62,
    // integer, number of list compliments received by the user
    "compliment_list": 37,
    // integer, number of note compliments received by the user
    "compliment_note": 356,
    // integer, number of plain compliments received by the user
    "compliment_plain": 68,
    // integer, number of cool compliments received by the user
    "compliment_cool": 91,
    // integer, number of funny compliments received by the user
    "compliment_funny": 99,
    // integer, number of writer compliments received by the user
    "compliment_writer": 95,
    // integer, number of photo compliments received by the user
    "compliment_photos": 50
}
```

Topic extraction will use both business.json and review.json.

To get the reviews of the Toronto restaurant. business.json contains location and service data which can be used to get business_id of restaurants in Toronto. business_id will further be used to get reviews from review.json.

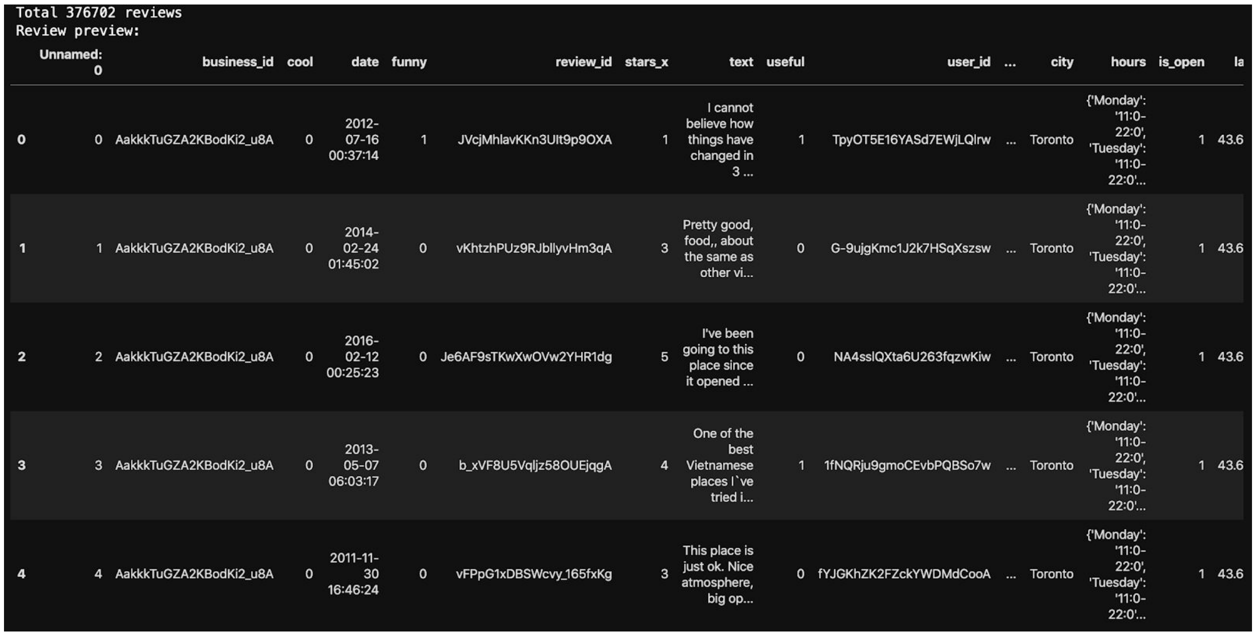
Data Wrangling

The dataset is pretty clean. However, the review data is very large. It will be read in chunk and merged with business data by business id.

Steps:

1. Get Toronto restaurants from business.json by categories and city
2. Read reviews from review.json by chunk and merge it with Toronto restaurants
3. Reset business_id and review_id as indexes
4. Rename columns

After wrangling, the data looks like this:

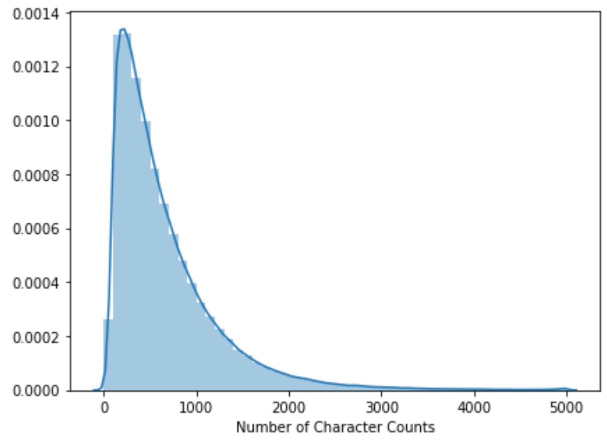


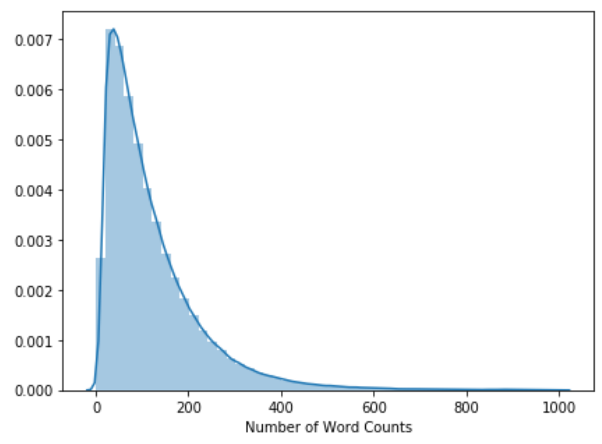
```
Total 376702 reviews
Review preview:
```

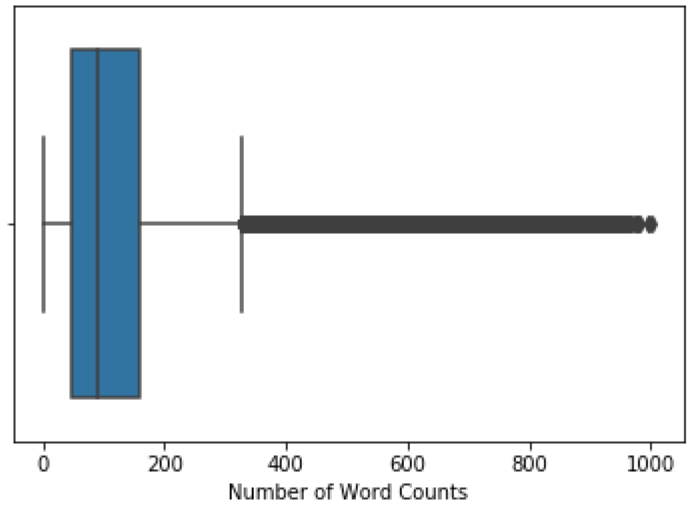
	Unnamed: 0	business_id	cool	date	funny	review_id	stars_x	text	useful	user_id	...	city	hours	is_open	la
0	0	AakkkTuGZA2KBodKi2_u8A	0	2012-07-16 00:37:14	1	JVcjMhlavKKn3UIt9p9OXA	1	I cannot believe how things have changed in 3 ...	1	TpyOT5E16YASd7EWjLQlrw	...	Toronto	{'Monday': '11:0-22:0', 'Tuesday': '11:0-22:0'...	1	43.6
1	1	AakkkTuGZA2KBodKi2_u8A	0	2014-02-24 01:45:02	0	vKhtzhPUz9RJbllyvHm3qA	3	Pretty good, food,, about the same as other vi...	0	G-9ujgKmc1J2k7HSqXszsw	...	Toronto	{'Monday': '11:0-22:0', 'Tuesday': '11:0-22:0'...	1	43.6
2	2	AakkkTuGZA2KBodKi2_u8A	0	2016-02-12 00:25:23	0	Je6AF9sTKwXwOVw2YHR1dg	5	I've been going to this place since it opened ...	0	NA4sslQXta6U263fqzwKiw	...	Toronto	{'Monday': '11:0-22:0', 'Tuesday': '11:0-22:0'...	1	43.6
3	3	AakkkTuGZA2KBodKi2_u8A	0	2013-05-07 06:03:17	0	b_xVF8U5Vqljz58OUEjqgA	4	One of the best Vietnamese places I`ve tried i...	1	1fNQRju9gmoCEvbPQBSo7w	...	Toronto	{'Monday': '11:0-22:0', 'Tuesday': '11:0-22:0'...	1	43.6
4	4	AakkkTuGZA2KBodKi2_u8A	0	2011-11-30 16:46:24	0	vFPpG1xDBSWcvy_165fxKg	3	This place is just ok. Nice atmosphere, big op...	0	fYJGKhZK2FZckYWDMdCooA	...	Toronto	{'Monday': '11:0-22:0', 'Tuesday': '11:0-22:0'...	1	43.6

Exploratory Data Analysis

1. What’s the average length of the review?



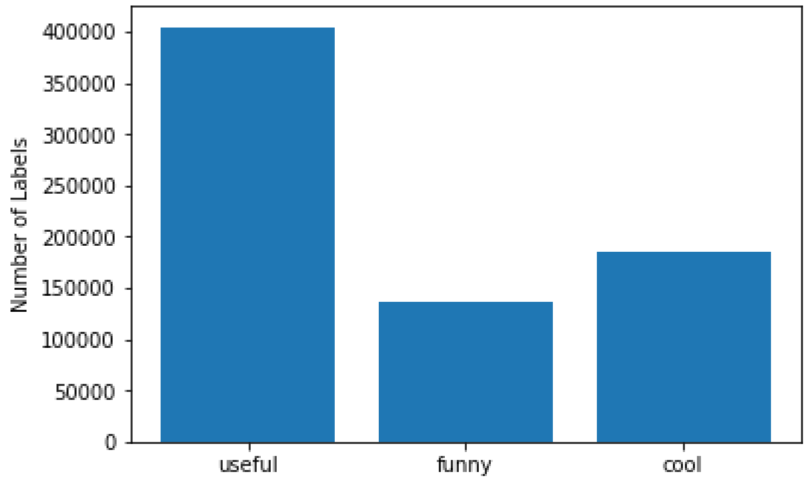




From the histogram plot, we can see that the number of word counts is not normally distributed and skewed to the right. The boxplot can also confirm this, we can see that there are a few outliers which are much larger than most of the counts.

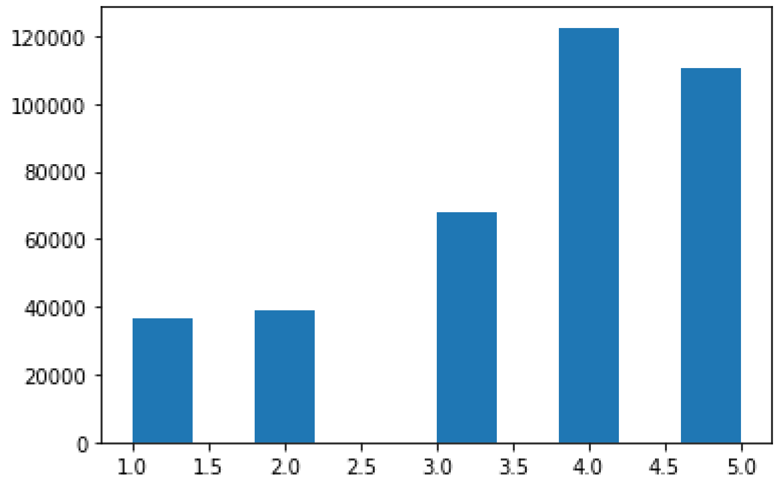
In this case, the median is a more meaningful way to estimate the average. So the average number of word counts in yelp review is 91.

2. There are three labels for reviews (useful, funny, cool), which one do people use more frequently?



The plot shows that people use useful label more often than the other two.

3. What is the distribution of the star given by customers?



From the plot, customers give more good stars (stars higher than 3).

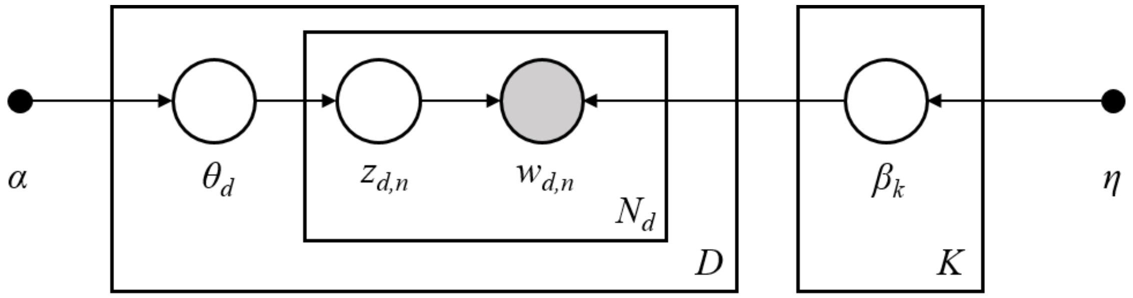
Topic Extraction By LDA

LDA

Latent Dirichlet Allocation (LDA) is applied in this project to abstract topics from the reviews.

LDA is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. It is also a topic model that is used for discovering abstract topics from a collection of documents.

The graphical model of LDA is a three-level generative model:



- The corpus is a collection of D documents.
- A document is a sequence of N words.
- There are K topics in the corpus.
- The boxes represent repeated sampling.

Feature Extraction

In order to perform machine learning on yelp reviews, review content need to be converted into numerical feature vectors. The reviews are converted into bags of words which are arrays of integers that represents the frequency of the word appearing in the reviews.

Parameter Tuning

The following parameters are tuned during the learning process of the model to get better results.

max_df and min_df

Document frequency (df) shows how frequent a specific word shows up in all the documents.

By specifying max_df, the vectorizer ignores all the words whose df is higher than the threshold. By specifying min_df, the vectorizer ignores all the words whose df is lower than the threshold.

If a word appears only in few documents, it probably doesn't contribute to understanding the topic of the documents. In a similar way, if a word appears in all documents, it can probably be removed.

As a result, max_df=0.8 and min_df=5 yield better and more meaningful topics.

ngram_range and token_pattern

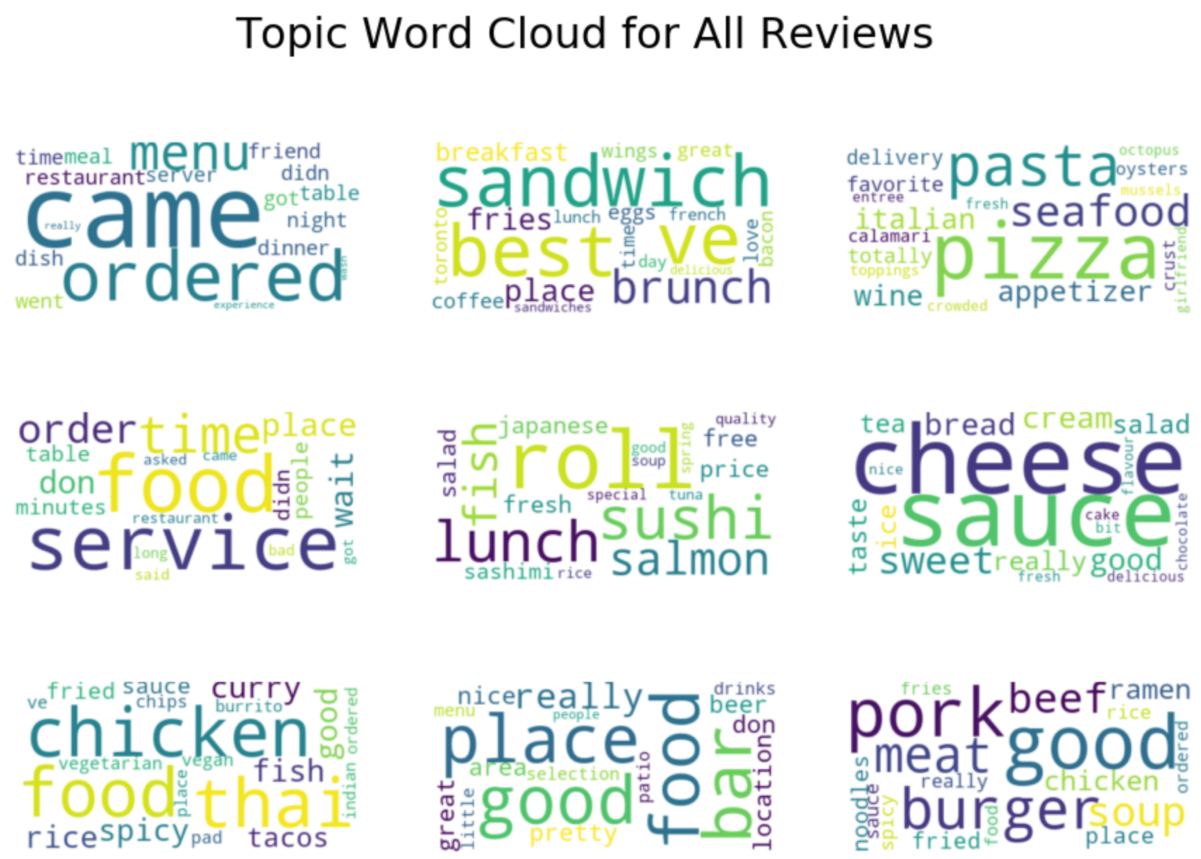
Applying bigram doesn't give us much improvement in understanding the topic. So (1, 1) is kept as the value of ngram_range.

TF-IDF (Term-Frequency Times Inverse Document-Frequency)

Some words in the text corpus give less meaningful information ('a', 'the' in English). Tf-idf can re-weight the count features into floating-point value.

However, after applying TF-IDF, there are some new and confusing words showing up in the result (criminal, aesthetics etc.). Tf-idf actually makes it harder to understand the topics. This may be caused by that those words are rarely showed up in the documents (reviews), so they are assigned with more weights than others.

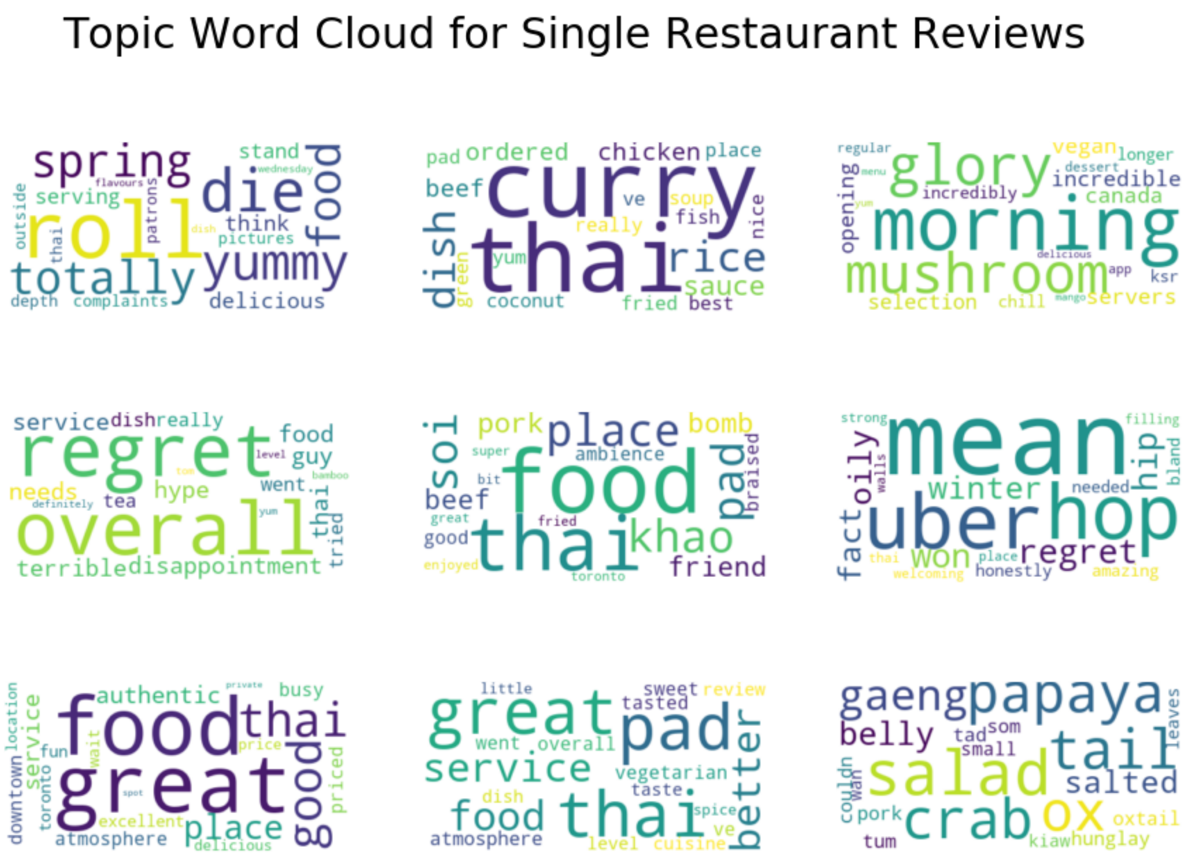
Results



- Topic 1: Pre-table service (negative)
- Topic 2: Breakfast and brunch restaurant
- Topic 3: Italian food
- Topic 4: Service time (negative)
- Topic 5: Japanese food
- Topic 6: Dessert
- Topic 7: Thai food
- Topic 8: Atmosphere (positive)
- Topic 9: Comfort food

These topics are useful for the customer. However, topics 1, 4 and 8 are particularly valuable for both customers and service providers (restaurants).

The reviews can be labeled with the extracted topics so they can be used as filters or for further classification.



From the above word cloud, we can see that the topic words for a single restaurant are much harder to interpret. In all these topics, there are two topics that are more interesting than others. The words from topic 1 suggest that the food is worth to die for. The words from topic 2 suggest that the service is terrible.

What’s Next

Due to the resource limitation, the analysis will stop here. However, there are lots of valuable things that can be built on top of this project left.

Star Prediction

Training a prediction model directly from the review content is difficult because of the high sparsity of bags of words. However, if the review text is converted into topic words, the number of features can be greatly decreased. Training the model with these new features may yield better prediction results.

Better and More Specific Topics

We can get better and more interpretable topic words from further tuning the parameters. For example, if we are not interested in the type of food in the restaurant, we can set words like sushi, pizza as stop words so they'll be removed from the features.

From the result of the single restaurant, we can see that the topic words from a single restaurant are more difficult to interpret. This can be improved by having fewer topics and tuning the alpha parameter of the LDA model.

Getting more specific topic words can also be achieved by limiting the features to specific bags of words. For example, to get the feelings' topic from the reviews, we can limit the features to the adjective words that express the feelings.

Service Clustering (Recommendation)

As we have the reviews labeled with the extracted topics, services can be clustered based on their reviews and the labels. A recommendation system then can be built upon this to recommend similar types of restaurants.