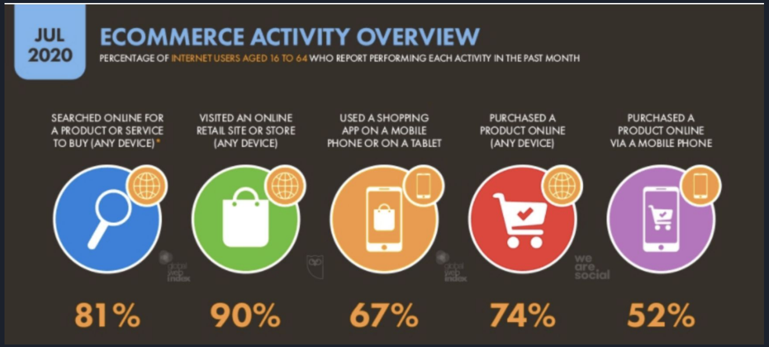
NLP Restaurant Search Engine

Steve Kim
Capstone Two Project
Springboard Data Science
Jan, 2020

Search Engine and Project Goal

- **Search engines are used everywhere (“google it”)**
  - E-commerce: Amazon, Target, Walmart, etc.
  - Online listings: Airbnb, Yelp, etc.
  - Entertainment: Netflix, HBO Max, Youtube, etc.
- **Users relies heavily on search engines**
  - Users searches for various information. For example, consumers searches for various products, software engineers looks for solution to the bug using google and stack overflow.
- **Capstone Project Goal**
  - Listing relevant restaurants when typing in specific ingredients or cuisine name (i.e. ‘galbi bulgogi).



Approach

- **Data Acquisition and wrangling**
  - Gather relevant csv files.
  - Web-scraping restaurants’ menu data.
- **EDA**
  - Visualize restaurants’ similarities based on its attributes.
  - Visualize common unigrams, bigrams, and trigrams.
  - Identify common words in each cuisines.
  - LDA topic modeling.
- **Building Doc2Vec and TF-IDF models**
  - Build three Doc2Vec models based on the following:
    - Yelp’s Review/Tips text values
    - Menu description
    - Menu title
  - Build TF-IDF model based on relevant keywords.

Target Audience

- **Restaurant Search Engine is useful for all parties**
  - Food enthusiasts and travelers looking for specific taste and/or atmosphere.
  - General public looking for places to dine or take-out.
- **Search engine can be changed for different applications**
  - Search engine can be converted to look for academic documents or any retail products.

Dataset

Yelp Dataset

- 8,021,122 reviews
- 209,393 businesses
- 1,320,710 tips
- Over 1.4 million business attributes.

Yummly Dataset

- 20 cuisines (Korean, Spanish, etc.)
- 36,568 ingredients

Allmenus (web-scraped)

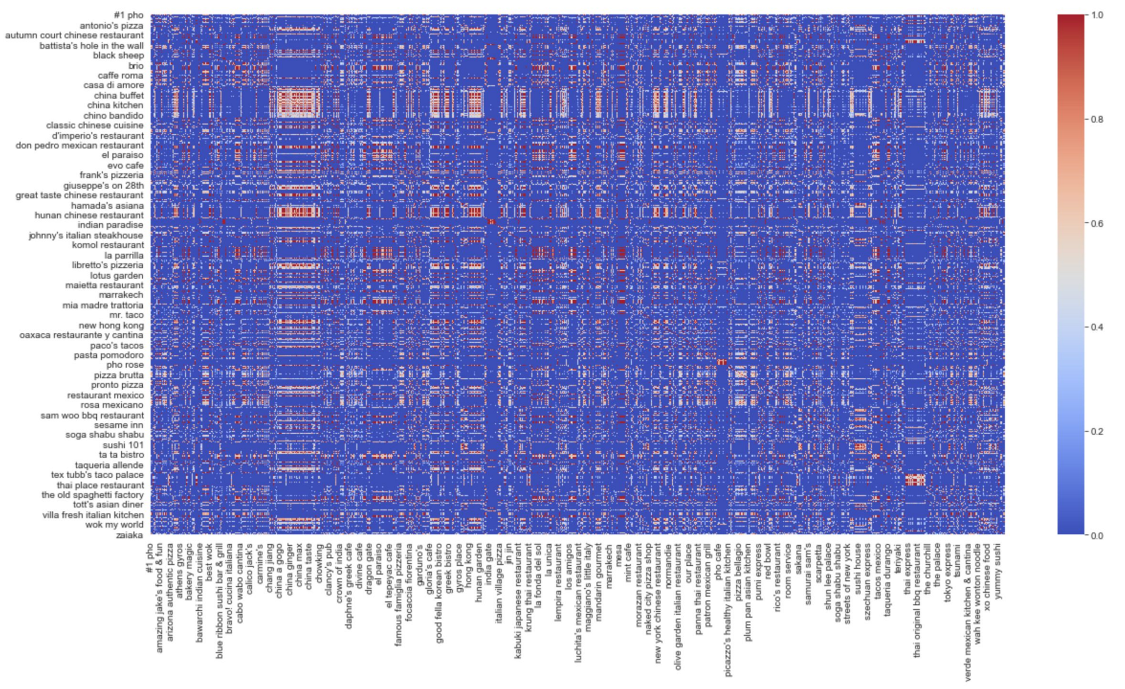
- 12,000+ restaurants
- 1,000,000+ menus

Data Wrangling

- **Text Pre-processing**
  - Expanded contractions, tokenized, lemmatized, lowercased, removed stop words and extra white spaces to all text values.
  - Filtered restaurant list down to 1,199 unique restaurants spanning across 22 U.S states.
  - Dropped rows with non-english language.
- **Data Quality Check**
  - Ensured there were no NaN values in any of the text columns.

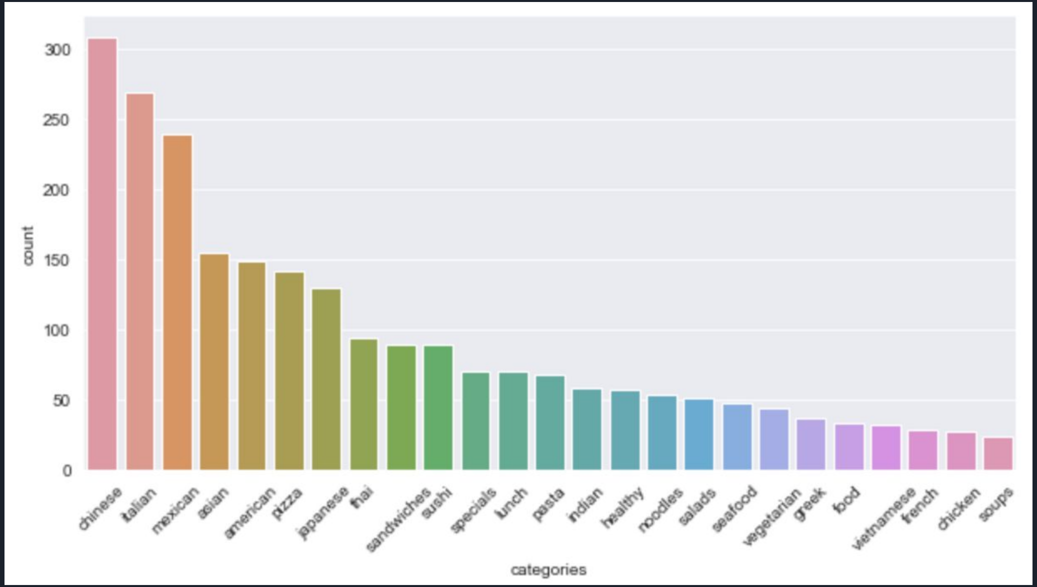
EDA - Similarities between Restaurants

Identified restaurants’ similarities using cosine similarity and Seaborn’s heatmap.



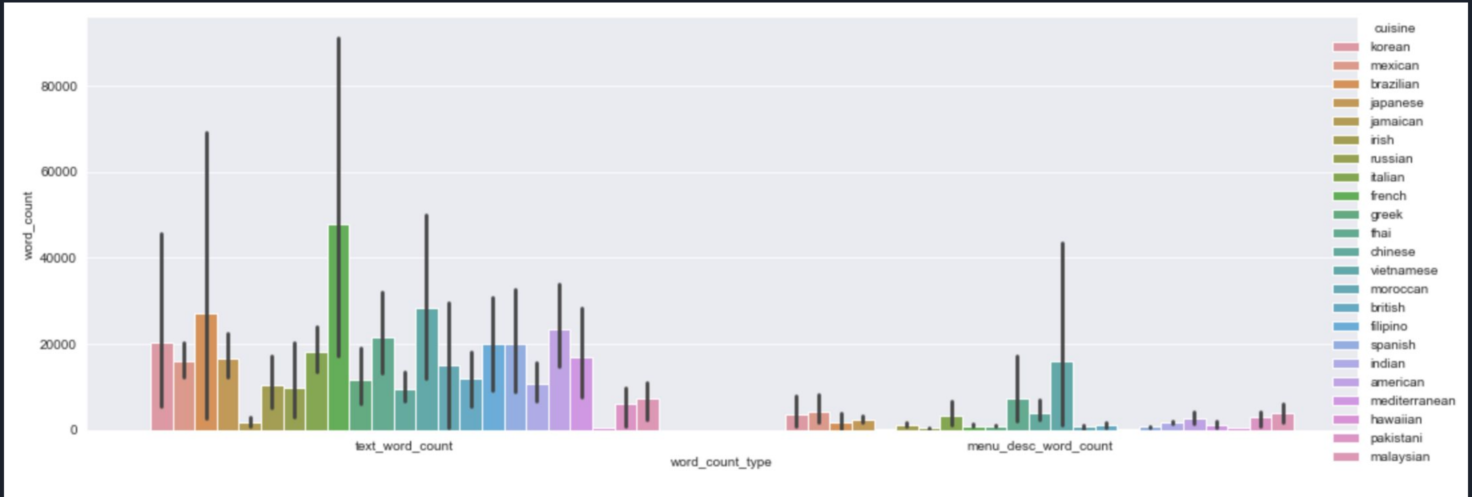
EDA - Top 25 Most Occurring Cuisines

Top 3 most occurring cuisines are ‘Chinese’, ‘Italian’, and ‘Mexican’.



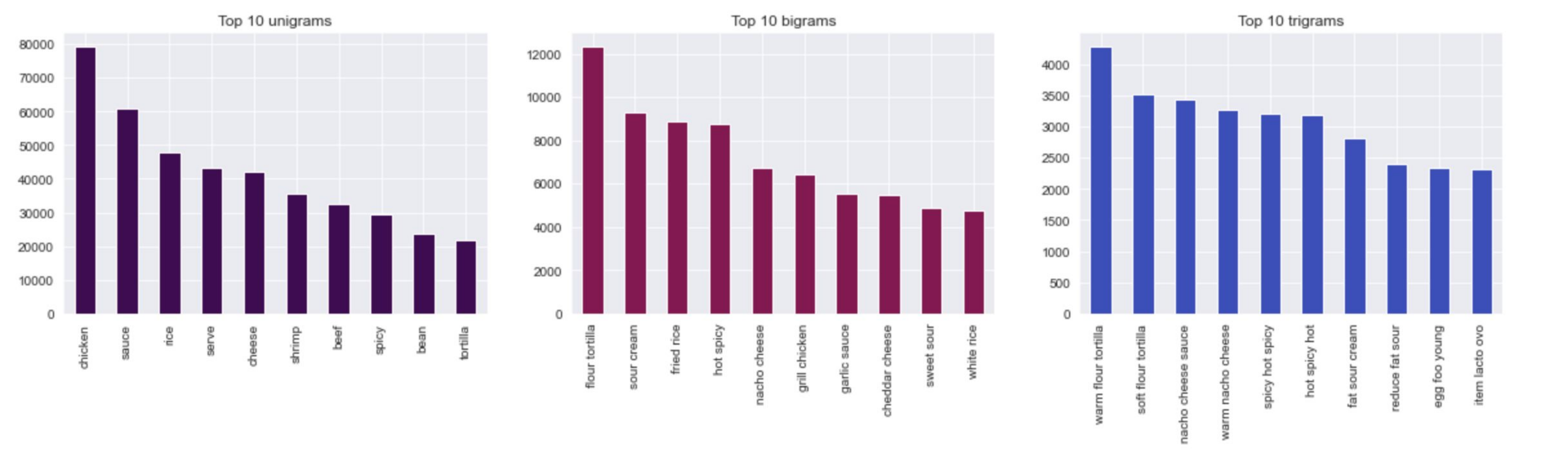
EDA - Word Count on each Cuisine

French cuisine has the most word count compared to other cuisines although not many French restaurants appeared in restaurants’ list. Furthermore, error bars are apparent on most cuisines which means that word counts are highly spread out from 3,000 to 100 words per document.



EDA - Distribution of Unigram, Bigrams, and Trigrams

The top 10 most common words from unigrams, bigrams, and trigrams are flour, spicy, tortilla, sauce, and milk related terms.



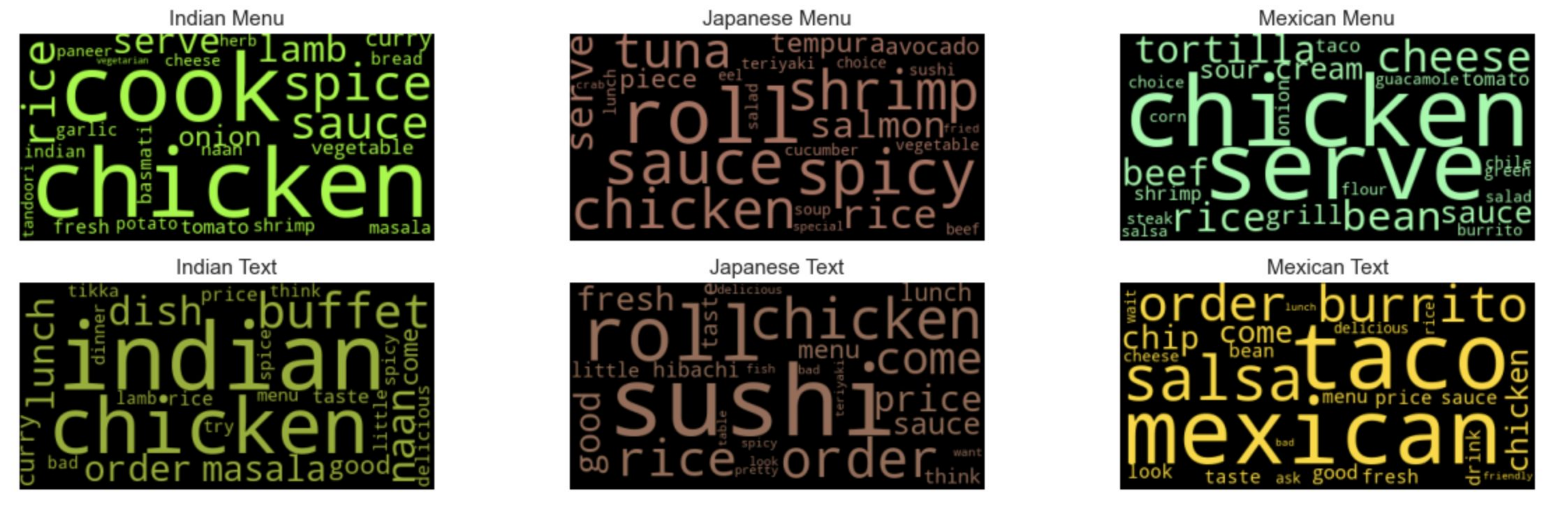
Top 10 unigrams

Top 10 bigrams

Top 10 trigrams

EDA - Top keywords of each Cuisines based on TF-IDF

Validated top keywords per cuisines. Japanese cuisine contained mostly fish or sushi related terms such as ‘tuna’, ‘roll’, ‘salmon’, ‘rice’, etc.



Indian Menu

Japanese Menu

Mexican Menu

Indian Text

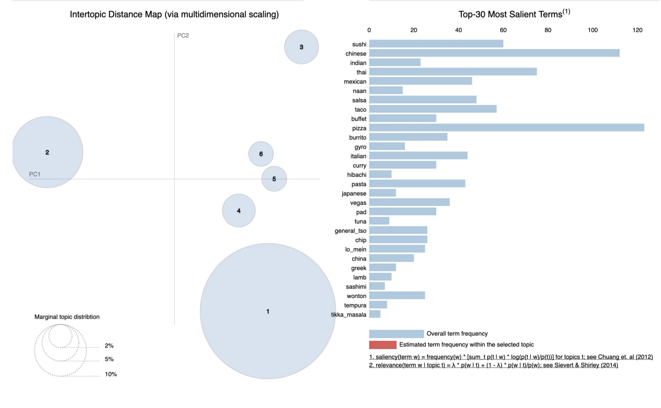
Japanese Text

Mexican Text

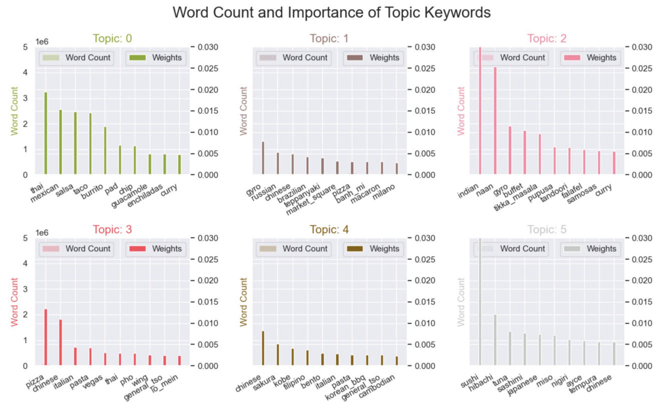
EDA - LDA Topic Modeling

Based on LDA model, six topics are the ideal topic count for Yelp's text values. However, due to general terminologies used throughout Yelp's review text, some were challenging understand more coherently.

- Topic 0 - Hispanic/Mexican
- Topic 1 - Traditional dishes or places(?)
- Topic 2 - Indian
- Topic 3 - Bread, Noodles, and starch (?)
- Topic 4 - Meat(?)
- Topic 5 - Japanese



1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

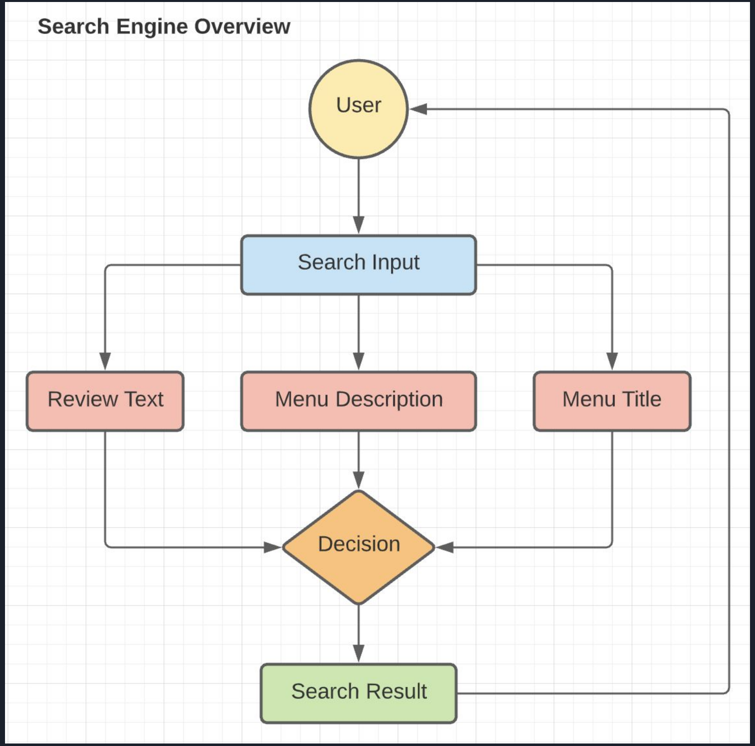

Word Count and Importance of Topic Keywords

NLP Search Engine - Overview

Three Doc2Vec models will be built and trained:

- Yelp’s Review/Tips text values
- Menu Description
- Menu Title

The image on the side is basic overview on how relevant restaurants will be created.

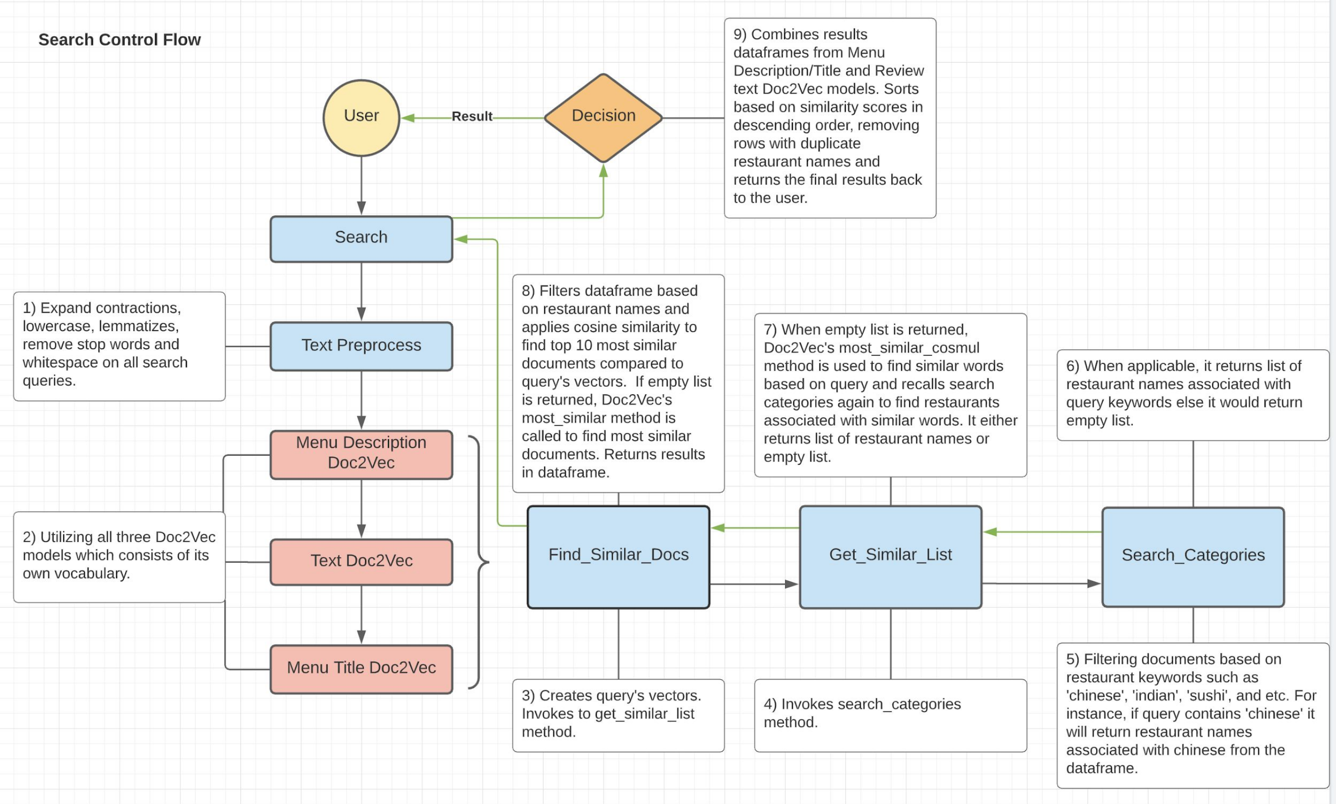


NLP - How relevant listings are populated

The idea behind building three Doc2Vec models is to consolidate search results which consists of most similar documents based on the user's search query using cosine similarity and built-in Doc2Vec's most_similar method. The top 10 documents will be chosen in one of following ways:

1. When there are common documents populated in all three results; the document will be added as part of the final result. For instance, when a restaurant named 'Panda Express' is shown in all three Doc2Vec results, Panda Express will be shown as part of the final result visible to the user.
   a. When similar restaurant names do not accumulate up to 10; the rest will be determined based on cosine similarity score.
2. When no common documents are found through three Doc2Vec results, all three results will be consolidated and results will be shown based on cosine similarity score in descending order.

NLP - Detailed Search Engine Control Flow



Search Control Flow

1) Expand contractions, lowercase, lemmatizes, remove stop words and whitespace on all search queries.

2) Utilizing all three Doc2Vec models which consists of its own vocabulary.

3) Creates query's vectors. Invokes to get_similar_list method.

4) Invokes search_categories method.

5) Filtering documents based on restaurant keywords such as 'chinese', 'indian', 'sushi', and etc. For instance, if query contains 'chinese' it will return restaurant names associated with chinese from the dataframe.

6) When applicable, it returns list of restaurant names associated with query keywords else it would return empty list.

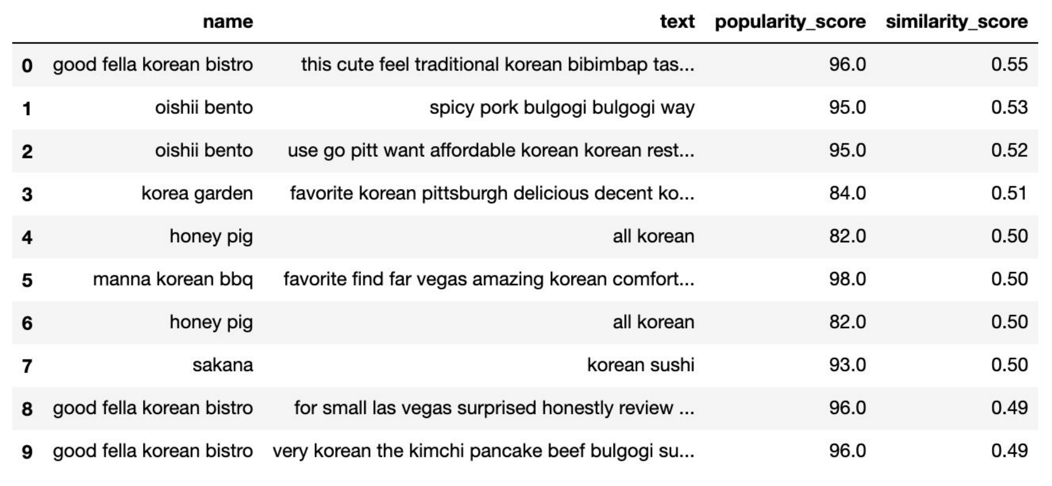
7) When empty list is returned, Doc2Vec's most_similar_cosmul method is used to find similar words based on query and recalls search categories again to find restaurants associated with similar words. It either returns list of restaurant names or empty list.

8) Filters dataframe based on restaurant names and applies cosine similarity to find top 10 most similar documents compared to query's vectors. If empty list is returned, Doc2Vec's most_similar method is called to find most similar documents. Returns results in dataframe.

9) Combines results dataframes from Menu Description/Title and Review text Doc2Vec models. Sorts based on similarity scores in descending order, removing rows with duplicate restaurant names and returns the final results back to the user.

NLP - TF-IDF Approach

Created similarity matrix between a list of keywords based on TF-IDF and used Gensim’s Matrix Similarity method in populating the result. The ‘korean bulgogi food’ search input returned the following result:



	name	text	popularity_score	similarity_score
0	good fella korean bistro	this cute feel traditional korean bibimbap tas...	96.0	0.55
1	oishii bento	spicy pork bulgogi bulgogi way	95.0	0.53
2	oishii bento	use go pitt want affordable korean korean rest...	95.0	0.52
3	korea garden	favorite korean pittsburgh delicious decent ko...	84.0	0.51
4	honey pig	all korean	82.0	0.50
5	manna korean bbq	favorite find far vegas amazing korean comfort...	98.0	0.50
6	honey pig	all korean	82.0	0.50
7	sakana	korean sushi	93.0	0.50
8	good fella korean bistro	for small las vegas surprised honestly review ...	96.0	0.49
9	good fella korean bistro	very korean the kimchi pancake beef bulgogi su...	96.0	0.49

Suggested Improvements

There are several ways to improve performances in identifying similar restaurants based on the user's search query.

- Using all of Yelp’s review text data instead of trimming down based on what was available on allmenus.com. At the end of my NLP project, I recognized that using a combination of LDA and Word2Vec/Doc2Vec, I could have found restaurants’ cuisine type without use of allmenus’ menu text data through topic modeling (clustering). Using this approach, may have provided more restaurants to search.
- Currently, the search engine populates based on similarity score only, the search result may be more relevant when populating list based on user’s location. For instance, if a user queried from Los Angeles, it would show restaurants near that area.
- Utilizing Spark or Dask in speeding up computational time in text pre-processing and populating search results.

Summary

- Search engine was built with the purpose of finding most similar restaurants based on user query whether it may be food, drink, or any restaurant's attributes.
- During the EDA process, we were able to differentiate and visualize text values based on cuisines (ex: 'Japanese', 'Chinese', 'French', and etc.).
- Created three Doc2Vec models based on Yelp's review text data and allmenus' menu data which provided coherent results in providing good results.
- The TF-IDF search model was created based on keywords which also provided coherent results but unlike Doc2Vec models it does not understand meaning of the words and semantic relationships between words and documents.

Acknowledgement

- Ajith Patnaik (Mentor)

Reference

- Detailed Project Report
- Data Wrangling & Text Pre-processing
- EDA
- Doc2Vec and TF-IDF Model Building
- Helper functions (search engine control flow)
- Project GitHub Repository