CS-598|Data Mining Capstone
Task 2: Yelp Data Cuisines Analysis
Net ID: shaukat2

Task Overview

The main goal of this task is to perform analysis on cuisines. This report provides detailed analysis on

- Visualization of Cuisine Map **(Task 2.1)**
- Improving Cuisine Map **(Task 2.2)**
- Clustering in Cuisine Map **(Task 2.3)**

Data Subset Details

I have used mainly business and reviews json files for this analysis. I performed join on these two datasets based on business id and category as Restaurants. Then I dropped irrelevant columns to keep the analysis clean and tidy. Analysis is performed on all open restaurants with review counts more than 9 throughout for this task.

Main Libraries Used

Gensim	Sci-kit Learn	matplotlib
Nltk	Scipy	Pandas
Numpy	Collections	seaborn

Data Pre-processing

Cuisines were extracted by applying "explode" function on categories and fetching unique ones. There are a total **197** categories / cuisines related to restaurants. I then gathered reviews for each cuisine/category type. These reviews were cleaned to remove newlines and stop words before storing. I also kept counts of these reviews. I then used these counts to select top 60 cuisines/categories.

Task 2.1: Visualization of the Cuisine Map

The main objective of this task is to gather similarities between different cuisines/categories and identify correlation.

Steps

To perform this task, following steps were taken.

- Applied TfidfVectorizer from sci-kit learn library with following settings
  - use_idf = False , min_df = 0.2, max_df = 0.5, max_features = 500
- Applied fit_transform method to generate sparse matrix
- Calculated cosine similarity on sparse matrix.
- Plot heatmap of this matrix as depicted in **Plot 1: Visualization of Cuisine Map without IDF**

Description

Initially, I used all 197 cuisines/categories, however heatmap produced by this was not visually readable. Therefore, I extracted top 60 cuisines/categories based on number of reviews. I applied TF-IDF Vectorizer with out IDF setting for this task to identify basic relationship between different types of cuisines.

Experiments

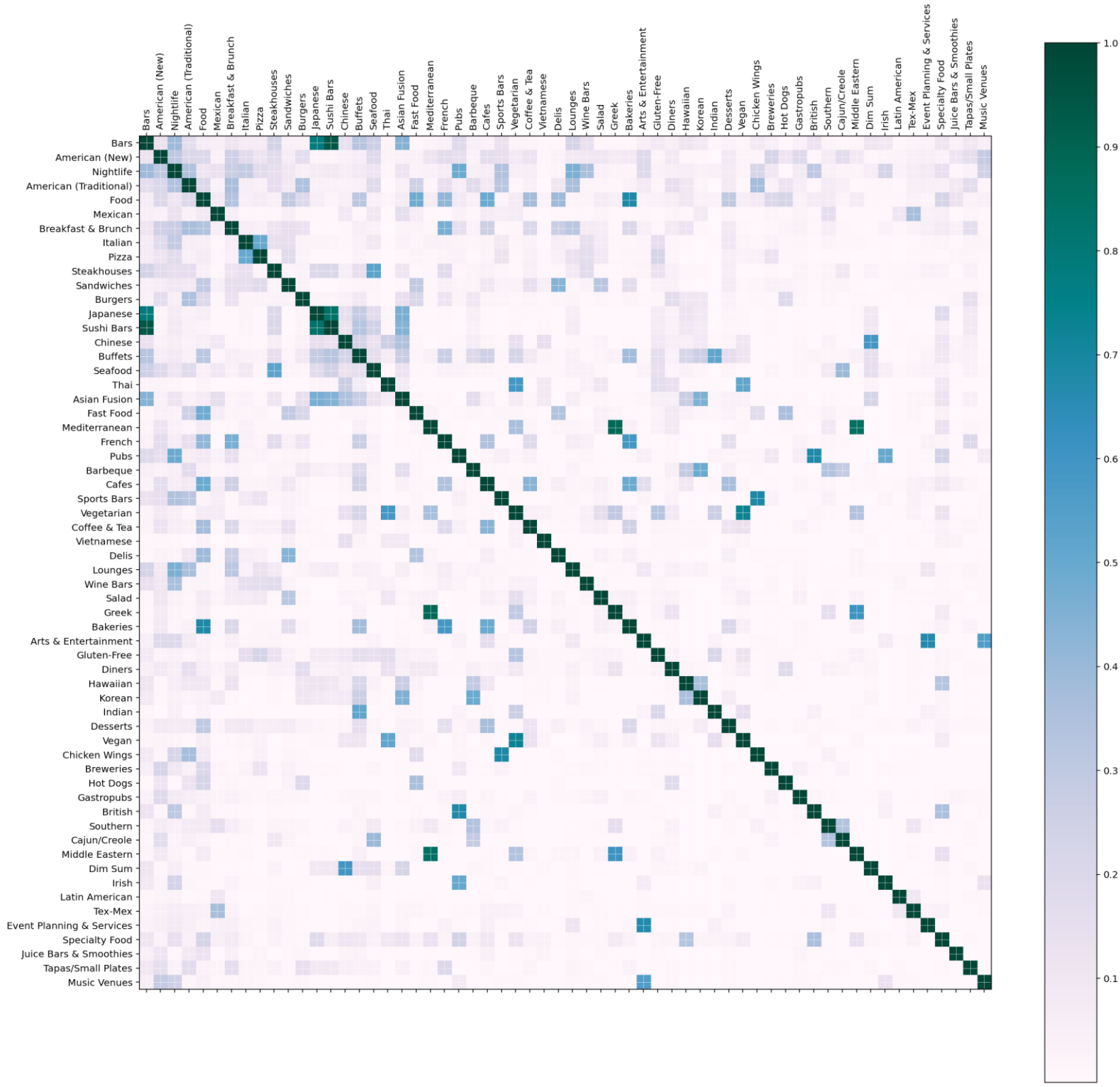
I played around with different setups for this task.

- I used Cosine, Euclidean and Manhattan distances as similarity function. However only Cosine performed better in this scenario.
- I also played around with different settings for TF-IDF Vectorizer input params like min_df, max_df and max_features. I got suitable results for
  - min_df as 20% threshold
  - max_df as 50% threshold
  - max_features as 500

Analysis

Referring to **Plot 1: Visualization of Cuisine Map without IDF** , we can see that cosine similarity function has performed optimally. We see strong relationship between

- Japanese and Sushi Bars.



Plot 1: Visualization of Cuisine Map without IDF

- Greek, Middle Eastern and Mediterranean
- Vegan and Vegetarian
- Dim Sum and Chinese

However, there are some cases where correlation is not string but does make sense for example,

- Korean and Hawaiian
- Desserts and Cafes

There are also some cases where the correlation is weak but it also doesn't make any sense. Like,

- Buffets and Bakeries

Overall, Similarity matrix generated with cosine similarity has generated a good similarity matrix.

Task 2.2: Improving Cuisine Map

Main objective of this task is to try and improve similarity measure between cuisines. I used couple of experiments to try and improve the cuisine similarity matrix.

Attempt # 1: Cosine Similarity with IDF

In this attempt, I enabled use_idf parameter of TF-IDF Vectorizer. This parameter uses inverse document frequency to weigh down quite frequent terms and scale up the infrequent ones. In other words, more importance is given to rare words.

Steps

To perform this task, following steps were taken.

- Applied TfidfVectorizer from sci-kit learn library with following settings
  - use_idf = True , min_df = 0.2, max_df = 0.5, max_features = 500
- Applied fit_transform method to generate sparse matrix
- Calculated cosine similarity on sparse matrix.
- Plot heatmap of this matrix as depicted in **Plot 2: Improved Cuisine Map with IDF**

Description

Top 60 cuisines/categories based on number of reviews were considered for this task as well. I applied TF-IDF Vectorizer with IDF setting enabled for this task to identify basic relationship between different types of cuisines.

Experiments

I played around with different setups for this task.

- I used Cosine, Euclidean and Manhattan distances as similarity function. However only Cosine performed better in this scenario.
- I also played around with different settings for TF-IDF Vectorizer input params like min_df, max_df and max_features. I got suitable results for
  - min_df as 20% threshold
  - max_df as 50% threshold
  - max_features as 500

Analysis

Let's first look at the relationships between cuisines that we discussed in Task 2.1. We see same strong relationship between

- Sushi Bars and Japanese
- Greek, Middle Eastern and Mediterranean
- Vegan and Vegetarian
- Dim Sum and Chinese

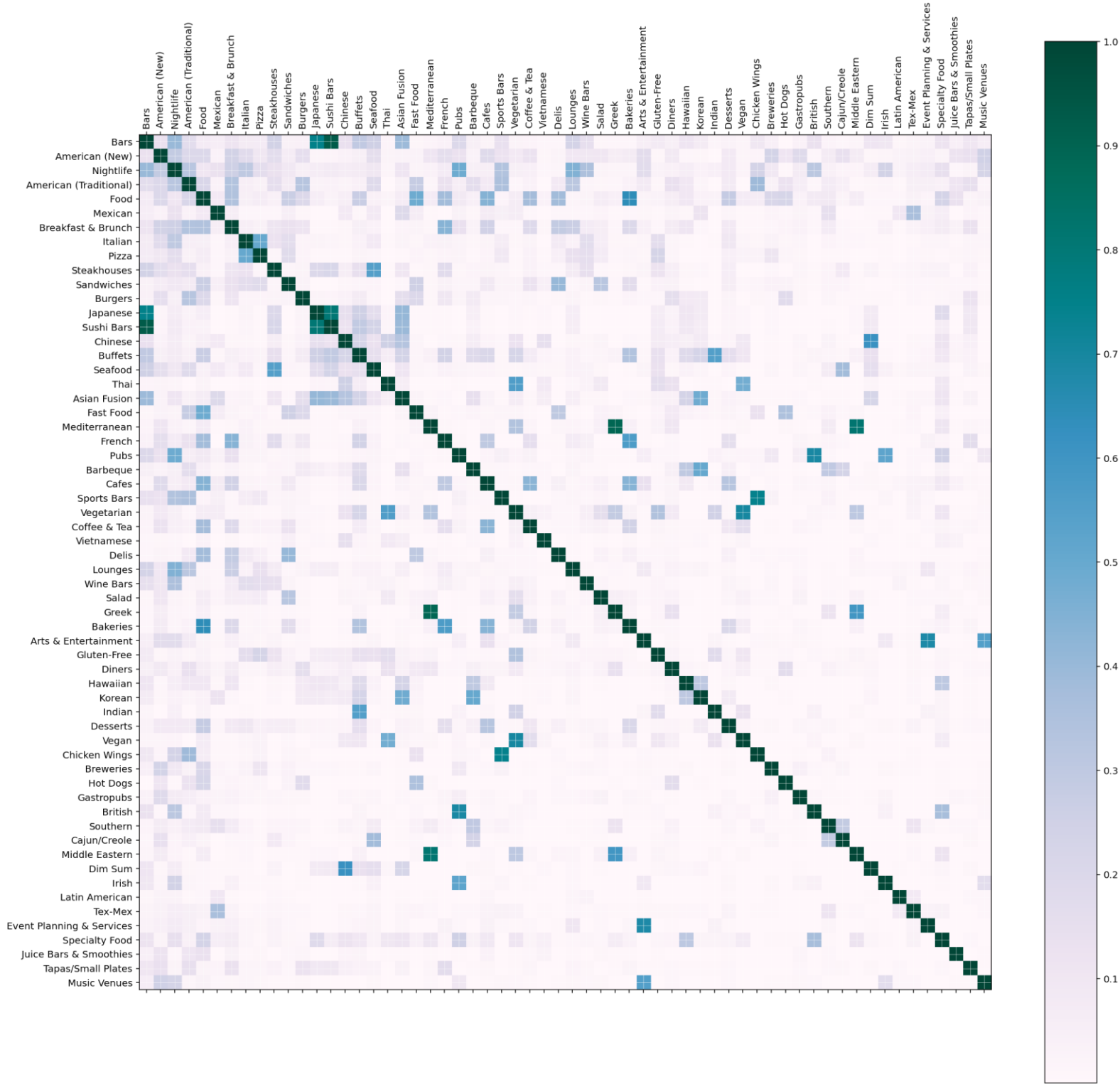
We also see same correlation between

- Korean and Hawaiian
- Desserts and Cafes

There also exist weak correlation between

- Buffets and Bakeries

Comparing IDF with No IDF, both return almost the same similarity matrix. There is no visible improvement between both heatmaps. This may be because these reviews are not written in professional language to be extremely topic specific hence use of relevant rare words is non-existent. This only proves that these reviews are written in casual day to day language by laymen.



Plot 2: Improved Cuisine Map with IDF

Attempt # 2: Cosine Similarity on all reviews with IDF enabled and LDA

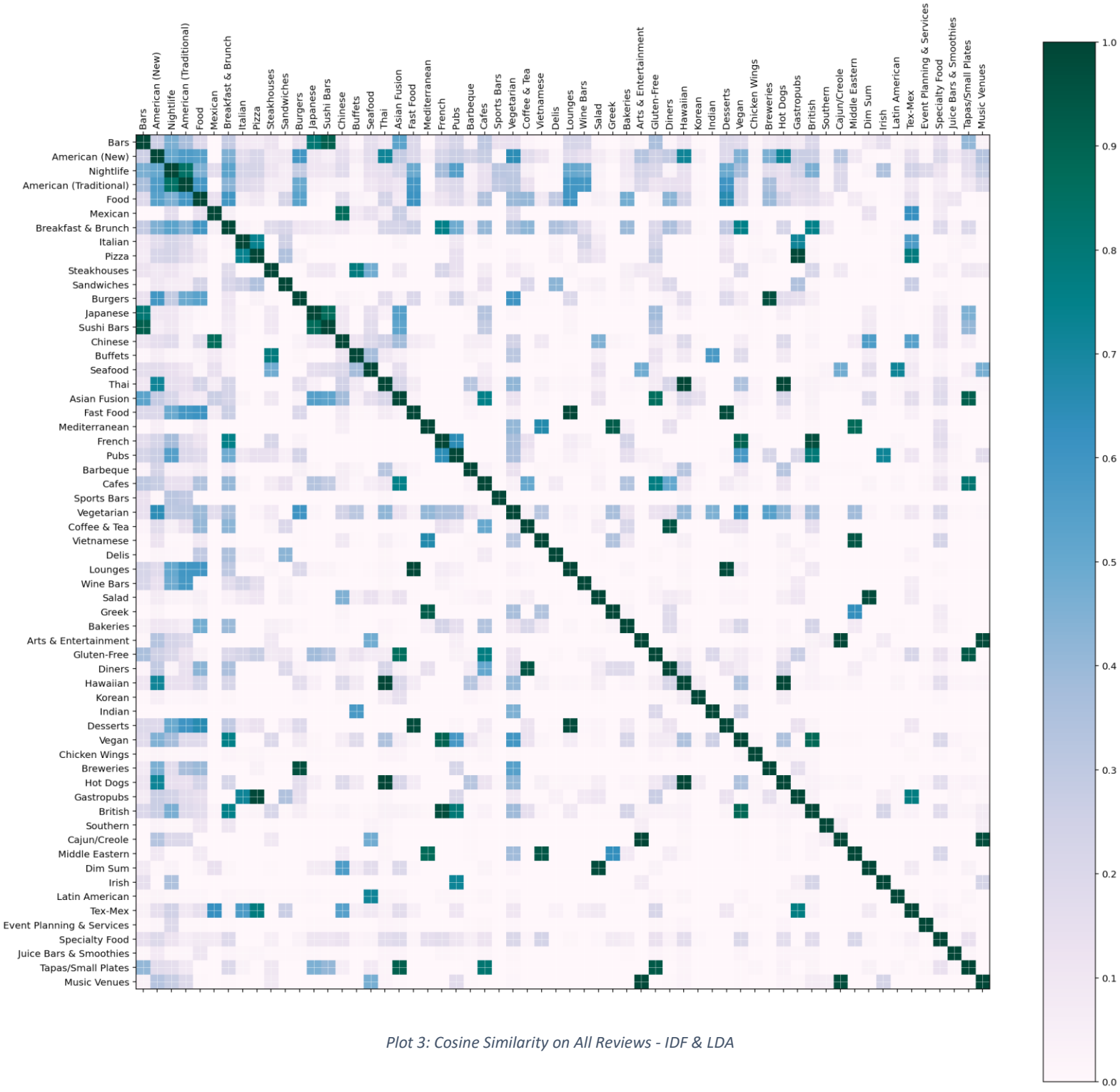
In this attempt, I enabled use_idf parameter of TF-IDF Vectorizer. Normally LDA performs great with embeddings like TF-IDF so I experimented with LDA along with IDF to see if it enhances similarity measures.

Steps

To perform this task, following steps were taken.

- Applied TfidfVectorizer from sci-kit learn library with following settings
  - use_idf = True , min_df = 0.2, max_df = 0.5, max_features = 1000
- Applied fit_transform method to generate sparse matrix
- Calculated cosine similarity on sparse matrix.

- Generated Corpus and dictionary based on the sparse matrix.
- Applied LDA with following settings
    - num_topics = 50, alpha = auto, eval_every = 5, iterations = 100, passes = 10
- Plotted heatmap of this as depicted in **Plot 3: Cosine Similarity on All Reviews - IDF & LDA**



Plot 3: Cosine Similarity on All Reviews - IDF & LDA

Description

Top 60 cuisines/categories based on number of reviews were considered for this task as well. I applied TF-IDF Vectorizer with IDF setting enabled and then LDA from Gensim library was applied for this task to identify and enhance relationship between different types of cuisines.

Experiments

I played around with different setups for this task.

- I played around with different settings for TF-IDF Vectorizer input params like min_df, max_df and max_features. I got suitable results for
    - min_df as 20% threshold
    - max_df as 50% threshold
    - max_features as 1000
- I also experimented with different parameters for LDA like num_topics but eventually settled for 50.

Analysis

Overall, we see a lot of strong correlations between different cuisines as compared to Plot 1 & Plot 2. Let's analyse how meaningful these relationships are. Let's first look at the relationships between cuisines that we discussed in previous 2 sections. We see that

- Sushi Bars and Japanese relationship has become even stronger
- Greek, Middle Eastern and Mediterranean is also pronounced
- Vegan and Vegetarian is prominent, but we also see Vegan correlated to British and Pubs which doesn't make a lot of sense.
- Dim Sum and Chinese are still strongly related; however, we see another strong relationship between Chinese, Mexican and Tex-Mex, which is interesting as these might be the cases where same restaurants provide Chinese and Mexican foods.
- Korean and Hawaiian relationship has faded.
- Desserts and Cafes relationship has also dissipated however we see strong ties of Desserts with Lounges and Nightlife and between Cafes and Tapas & Asian Fusion.
- Buffets and Bakeries has also dissipated while we see strong link between Buffets and Indian & Steak houses.
- We also see many new strong similarities like between Tapas and Gluten Free, British and Pubs, Tex Mex with Pizza etc.

All in all, the similarity between different cuisines has greatly improved. We have some suspicious pairings as well but they're quite minimal if compared with good and meaningful similarities.

Attempt # 3: Cosine Similarity on reviews one by one with IDF enabled and LDA

In this attempt, most settings were kept same as Attempt # 2. The only difference is that in this attempt cosine similarity is calculated on each review text based on topics generated by LDA.

Steps

To perform this task, following steps were taken.

- Applied TfidfVectorizer from sci-kit learn library with following settings
    - use_idf = True , min_df = 2, max_df = 0.5, max_features = 1000
- Applied fit_transform method to generate sparse matrix
- Generated Corpus and dictionary based on the sparse matrix.
- Applied LDA with following settings
    - num_topics = 50, alpha = auto, eval_every = 5, iterations = 100, passes = 10
- Calculated Similarity between each review and topics using similarity matrix
- Plotted heatmap of this as depicted in **Plot 4: Cosine Similarity One by One Review - IDF, LDA**

Description

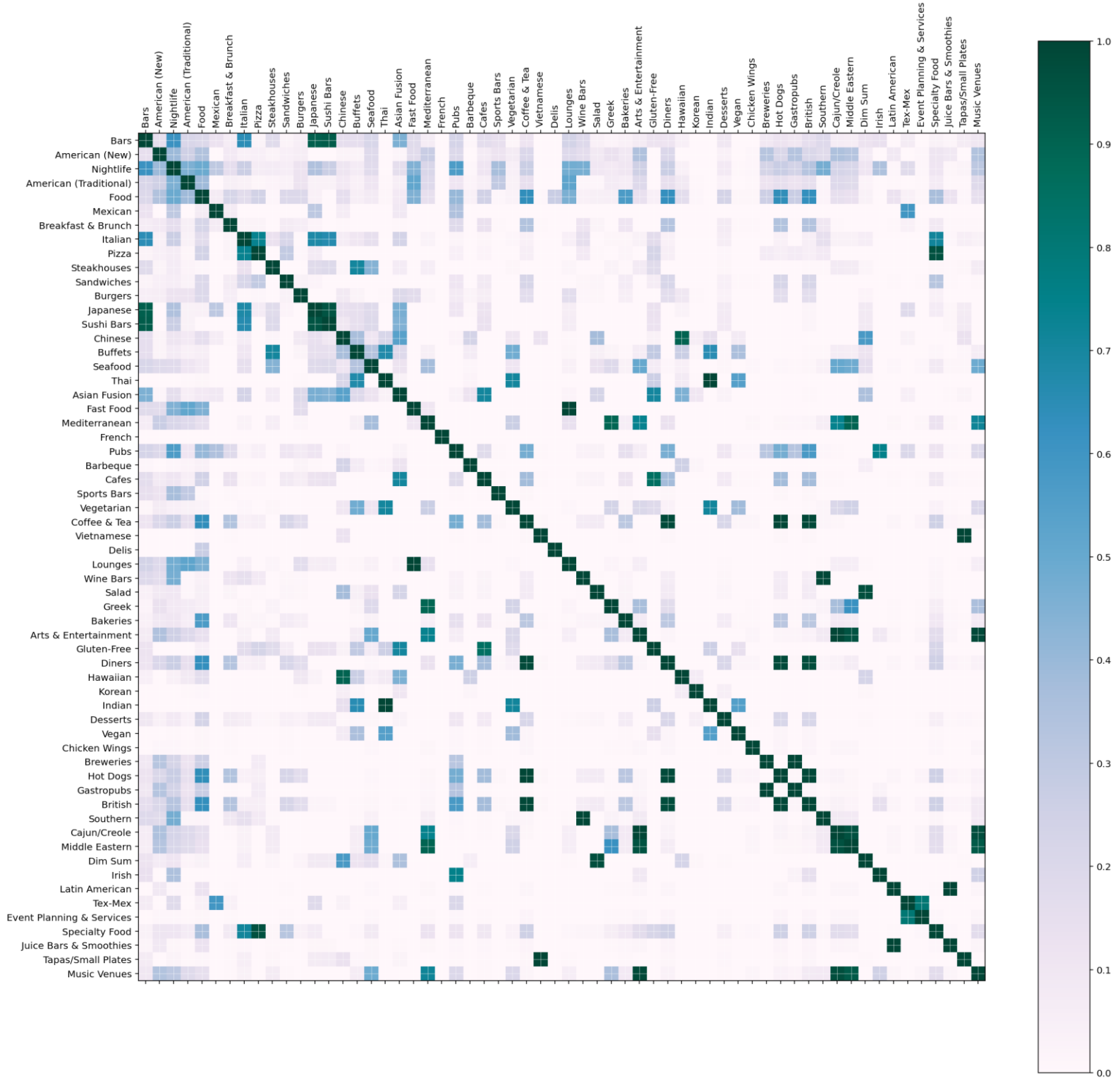
Top 60 cuisines/categories based on number of reviews were considered for this task as well. I applied TF-IDF Vectorizer with IDF setting enabled and then LDA from Gensim library was applied for this task to identify and enhance relationship between different types of cuisines. Cosine similarity was calculated for each review text and then aggregated to generate similarity matrix.

Experiments

I played around with different setups for this task.

- I played around with different settings for TF-IDF Vectorizer input params like min_df, max_df and max_features. I got suitable results for
    - min_df as 2
    - max_df as 50% threshold
    - max_features as 1000

- I also experimented with different parameters for LDA like num_topics but eventually settled for 50.



Plot 4: Cosine Similarity One by One Review - IDF, LDA

Analysis

Overall, we see a lot of strong similarities between different cuisines but we also see a lot cleaner plot as compared to plot 3. Let's first look at the relationships between cuisines that we discussed in previous sections. We see that

- Sushi Bars and Japanese have reached the maximum similarity value 1.
- Greek, Middle Eastern and Mediterranean is still pronounced. But now we also see a strong relevance between Mediterranean, Middle Eastern and Cajun/Creole cuisines which actually have strong influence on each other.

- Vegan and Vegetarian is prominent, but we also see Vegan correlation shown in Plot 3 between British and Pubs has dissipated. We now see strong correlation between Vegetarian and Indian which makes a lot of sense.
- Dim Sum and Chinese are still strongly related; however, we also see strong correlation between Dim Sum and Salad, both are considered healthy foods.
- We see Cafes relationship with Asian Fusion and Arts and Entertainment is quite prominent.
- Buffets has strong links to Indian, Steak houses and Vegan.
- We also see many new strong similarities of Pubs with British, Irish, Breweries, Coffee and Tea which makes a lot of sense.

This attempt has overall best results and the similarity between different cuisines has greatly improved. Most of the suspicious pairings we saw in plot 3 have vanished as well.

Task 2.3: Clustering in Cuisine Map

Main goal for this task is to implement clustering algorithms to identify and visualize related cuisines. I have used multiple approaches to implement clustering algorithms. I used similarity matrix generated in the Attempt # 3 for this task.

K-Means Clustering

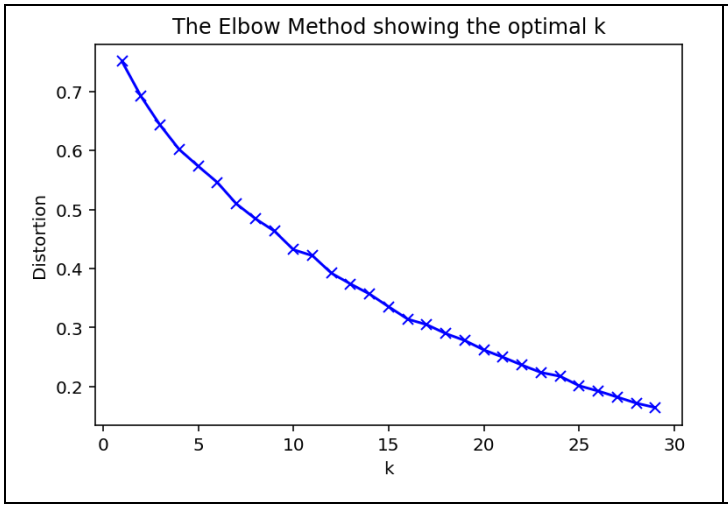
For implementation of K-Means Clustering, I chose similarity matrix obtained in Section "Attempt # 3". First of all, I implemented Elbow Method to identify optimal K.

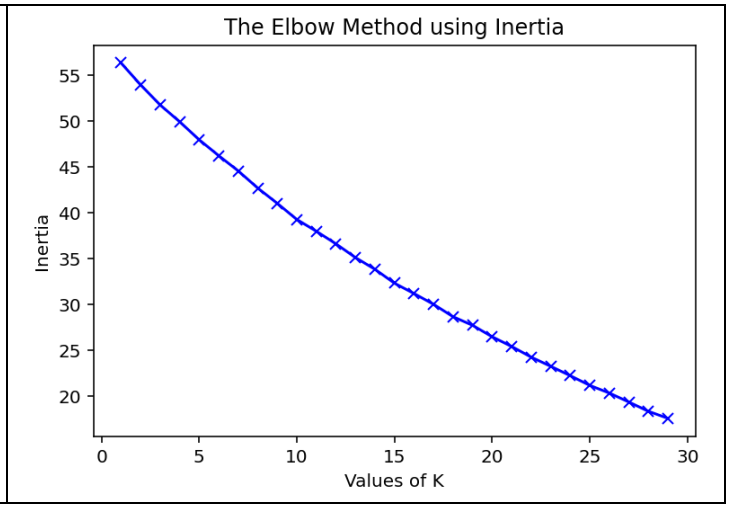
Elbow Method to find Optimal K

For its implementation, I used following settings to calculate distortions and inertia.

- Init= k-means++, n_clusters = range(1,30), n_init = 500
- Distortions calculated based on cosine

Since we have only 60 categories/cuisines, so this method didn't help a lot in figuring out optimal k. However, we can see a slightest ever bent around 10-15 k in "The Elbow Method showing the optimal k"





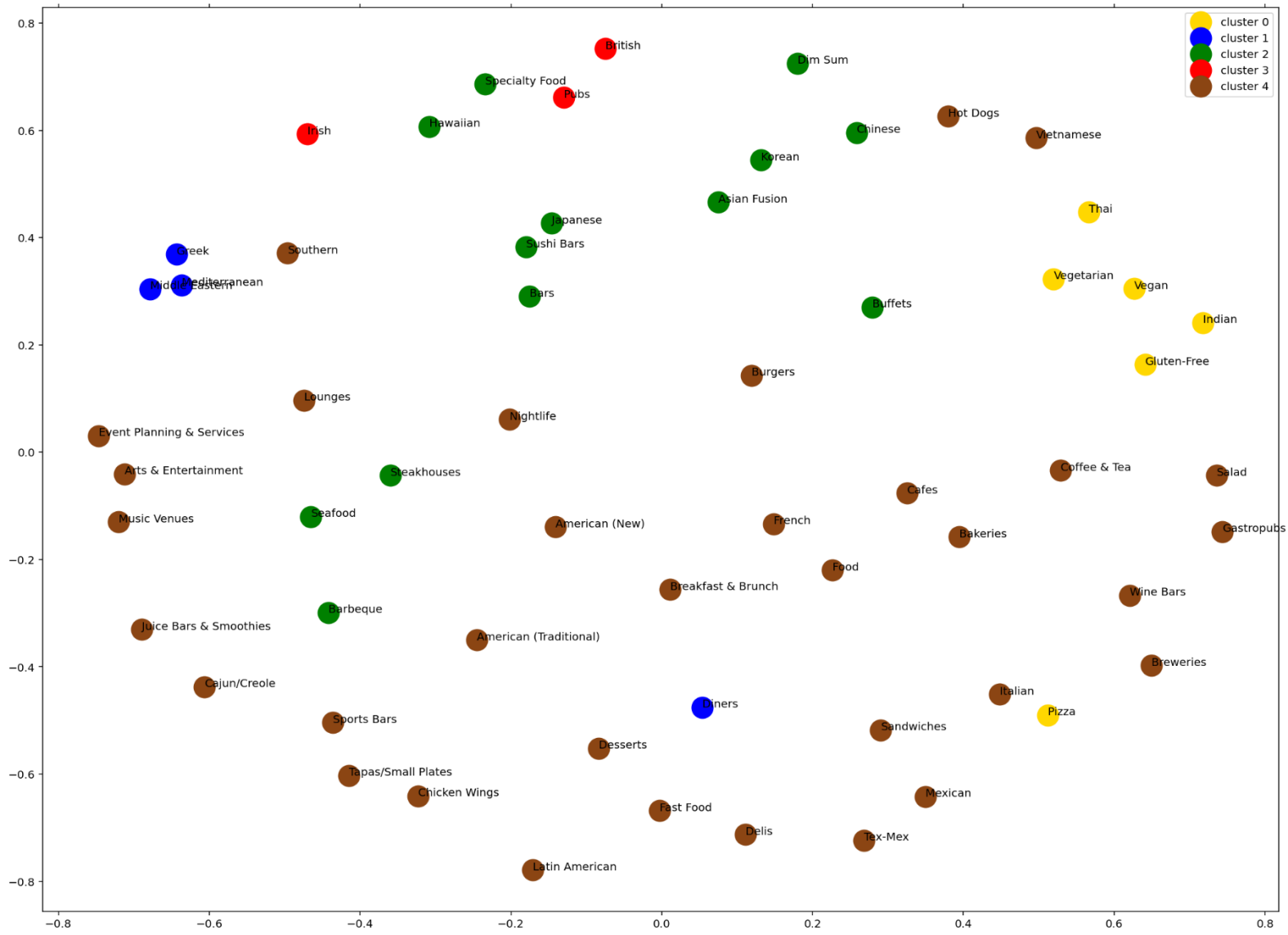
K-Means++ with num_clusters = 5 and 15

I implemented a K-Means++ for this task. K-Means++ is a smart centroid initialization technique and the rest of the algorithm is the same as that of K-Means. I used following settings.

- init='k-means++', n_clusters=num_clusters, n_init=100

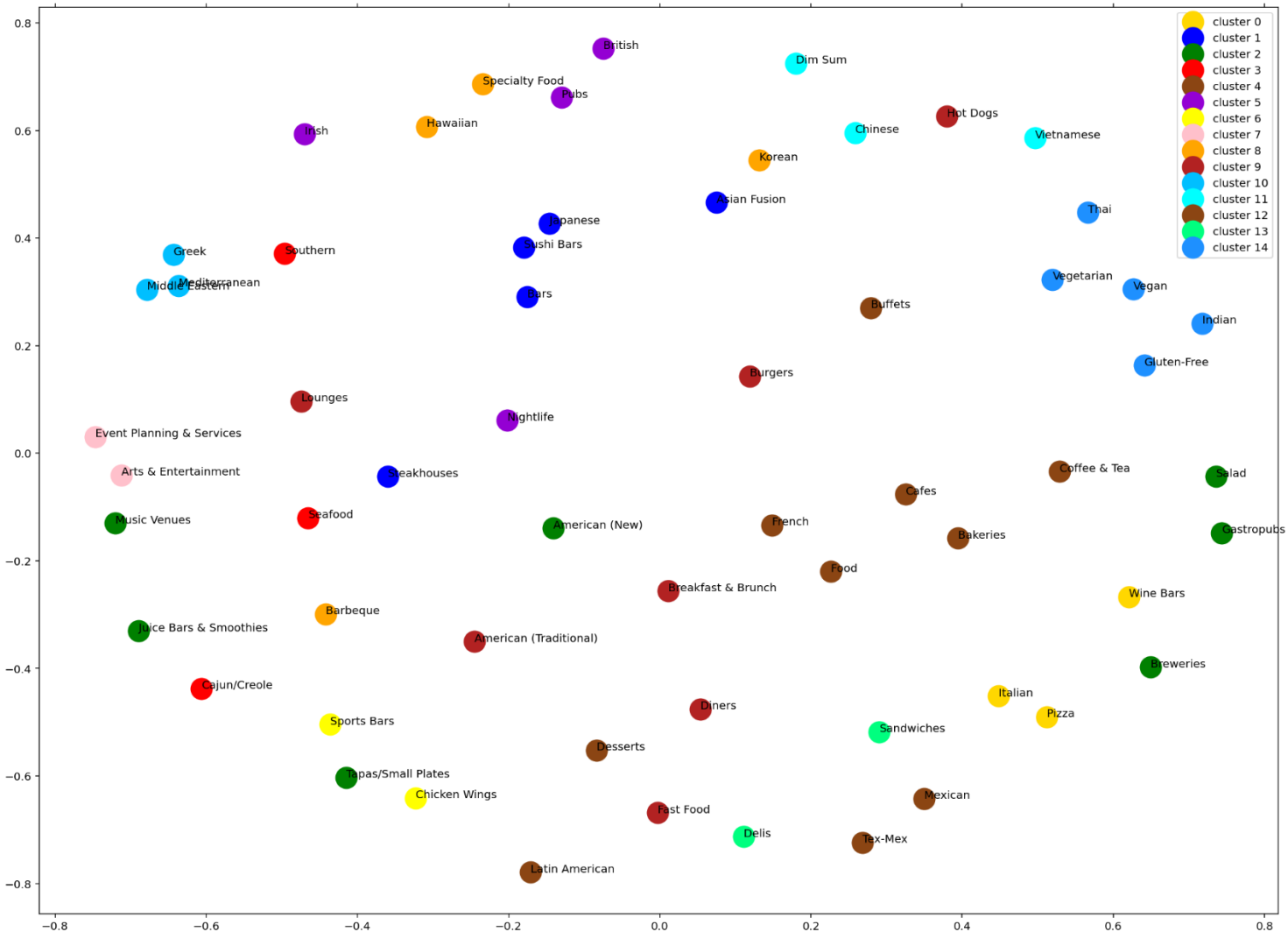
I then generated cluster visualization using matplotlib.

- For num_clusters = 5, **Plot 5: K-Means++ for k=5**
- For num_clusters = 15, **Plot 6:K-Means++ for k = 15**



Plot 5: K-Means++ for k=5

For number of clusters set to 5, We can see an optimal result of clustering with still room for improvement. We also see outliers like for cluster 0, pizza doesn't make a lot of sense in this cluster. Same way Diners may not make a lot of sense in cluster 1. We also see that since number of clusters is quite low so one of the clusters i.e. cluster 4 has been assigned a lot of cuisines which are not closely related.



Plot 6:K-Means++ for k = 15

For number of clusters set to 15, we see some very good clusters like cluster 10, cluster 0, cluster 5 etch are very well put and have strong relevance with data points in the cluster. This shows that improving number of clusters has improved the clustering mechanism and has also efficiently place highly related cuisines/categories in same cluster.

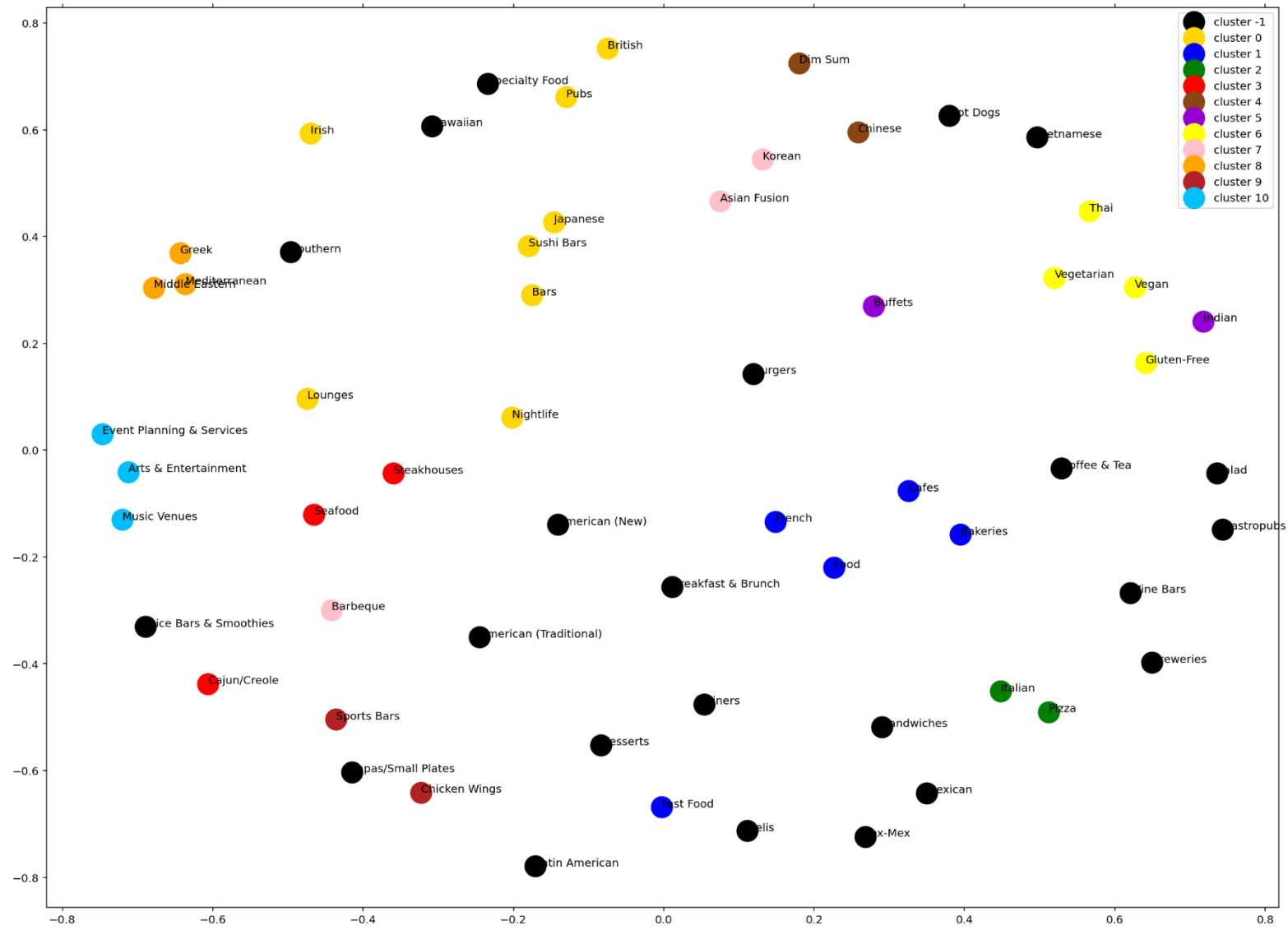
DBScan Clustering

Next clustering algorithm that I am going to use is DBScan. This algorithm is actually slower than K-Means but since we have limited number of categories/cuisines, we can easily utilize this algorithm. One other key point to note is that DBScan doesn't need number of clusters to be defined early. This algorithm also rejects data points it evaluates to be noise.
I used following settings.

- eps=1, min_samples=2
- eps is the maximum distance between two samples for one to be considered as in the neighbourhood of the other. After quite hit & trial its value was set to be 1.
- min_samples is the number of samples in a neighbourhood for a point to be considered as a core point.

After that I generated a plot using matplotlib **Plot 7: DBScan.**
We can see that it only generated 11 clusters and most of the points were assigned to cluster -1 which is a default cluster. Looking at Plot 6 and Plot 7, it is clear that K-Means++ performed better than DBScan. Some of the clusters in DBScan are mis formed for example Cajun/Creolo does not make a lot of sense in cluster 3.

**K-Means and DBScan Clustering Visualizations are created with the help of Multidimensional Scaling (MDS) which is is a technique that creates a map displaying the relative positions of a number of objects, given only a table of the distances between them.*



Plot 7: DBScan

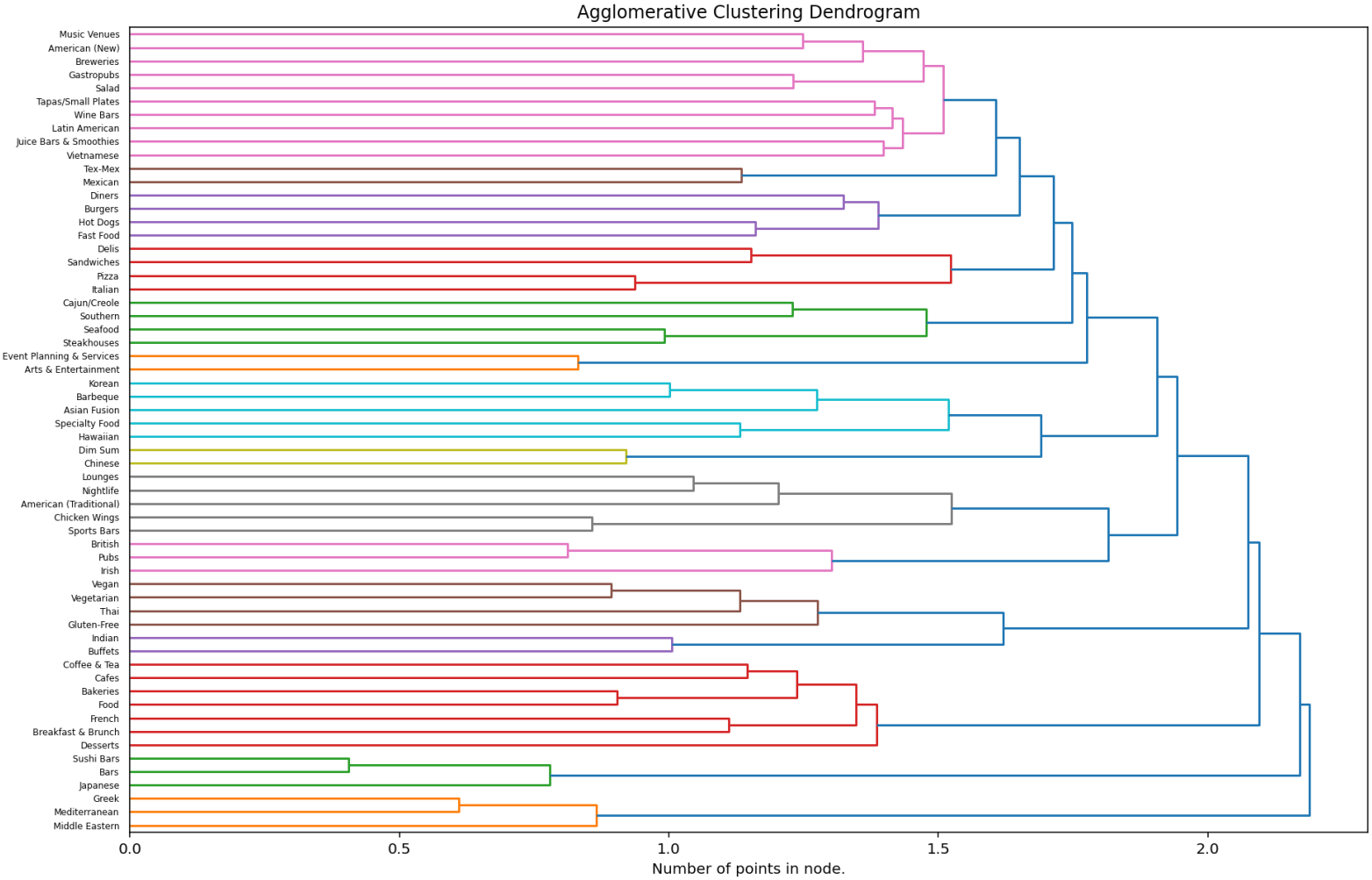
Agglomerative Clustering

This is a very popular technique used in data mining to perform hierarchical clustering. The algorithm starts by treating each object as a singleton cluster. Next, pairs of clusters are successively merged until all clusters have been merged into one big cluster containing all objects. It is also called Bottom Up Clustering.
For its implementation I used sci-kit Learn library and used following settings.

- distance_threshold=0, n_clusters=None, linkage="ward"
- Ward's method for linkage is used to calculate the similarity between clusters.

I then plotted a dendrogram using scipy. Dendrogram nicely depicts the hierarchical relationship between these categories/cuisines.

- We can see that Greek and Mediterranean are on same level and have Middle Eastern as their parent.
- We can also see that Mexican and Tex-Mex are assigned to same cluster which makes a lot of sense.
- Same similarities can be seen between Vegan and Vegetarian and also British and Pubs.



Plot 8: Agglomerative Clustering Dendrogram

Overall, each clustering algorithm provides a different kind of insight in to cuisine/category similarity. K-Means++ was very efficient in showing relevant and related data points clubbed in to clusters while dendrogram based on hierarchical clustering helps see the hierarchical relationship between different cuisines.