JSC270, Winter 2020 - Prof. Chevalier

Exploration of bussiness in Yelp dataset

1. Introduction

Yelp is a local-search service powered by crowd-sourced review forum run by an American multinational corporation. It develops, hosts and markets Yelp.com and the Yelp mobile app, which publish crowd-sourced reviews about local businesses, as well as the online reservation service Yelp Reservations

```
                                                                    -- Adapted from Wikipedia
```

Imaging that, one day, you feel hungry and feel tired to make food by your own. So you decide to go outside and find a wonderful restaurant with delicous food. While you struggle on deciding which restaurant to eat, you friend come with a mobile phone showing all restaurants nearby, you look at the review platforms and click on a 5 stars restaurants and read comments from previous diners. The compliment from previous customers help you make decision to go to this place This happens almost everyday in my life.

Nowadays, review platforms has played an huge impact on people's decision making. From customer's point of view, those reviews help us make more informed decisions and lead to a better experience. For all restuarants and local business, it is clear that reviews have huge impact on business revenue since customers tend to choose restuarants with better review, and one single one-star review significantly damage business's reputation, and might lead to a loss of various potential customers.

Therefore, how can business improve customer experience and manage to get better review ? This report mainly focus on restaurants in GTA area and aims to decide which features a good restaurants should have by analysing various information about restaurants and corresponding reviews provided by yelp, specifically, which features are customer tends to like. Hope this data analysis could provide implication for all restaurants owners in great Toronto and help them to improve thier service.

Data

This dataset is a subset of Yelp's businesses, reviews, and user data. It was originally put together for the Yelp Dataset Challenge which is a chance for students to conduct research or analysis on Yelp's data and share their discoveries.

Please find the data here: https://www.yelp.com/dataset/challenge.

By The Term Of Use, you should notice the data could only be used for academic popurse such as developing academic project for courses in school. The data might only be used by me for academic purpose without violating the law and regulation. I understand I shall not rent, lease the sublices of the data, shall not thansfer the data product to others for commercial purpose.

The dataset contains various information about around 190000 local businesses across 10 metropolitan areas and corresponding user's reviews for each business.

Attributes of business data

- business_id: 22 character unique string business id,
- name: the business's name
- address: address of the business
- city: city of business
- state: state code for business
- postal code: postal code for business
- latitude: latitude of business
- longitude: longitude of business
- stars: star rating for business
- review_count: number of reviews for business
- is_open: 0 or 1 for closed or open, respectively
- categories: categories for business
- attributes: user given attributes for business

In this report, I will performing a data analysis on both business data and reviews data.

Firstly, I imported the required libraries and load data for business.

```
import pandas as pd
import numpy as np
import json
import seaborn as sns
import matplotlib.pyplot as plt
import string
```

```
import nltk
import folium
import warnings
import requests
import math
from nltk.corpus import stopwords
from geopy import distance
from folium.plugins import HeatMap
from geopy.geocoders import Nominatim
from wordcloud import WordCloud, STOPWORDS
from folium.features import CustomIcon
warnings.filterwarnings('ignore')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/shangluy/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
True
```

```
# loading data on for business
temp = []
with open("yelp_dataset/business.json") as file:
    for line in file:
        temp.append(json.loads(line))
business = pd.DataFrame(temp)
```

Data cleaning for business data

As, you can see, the data is pretty tidy. Firstly, I will do is droping all data with missing category and drop all columns I don't need for future data analysis

```
business.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	Unnamed: 0	address	attributes	business_id	categories	city	hours	is_open	latitude	lo
0	0	2818 E Camino Acequia Drive	{'GoodForKids': 'False'}	1SWheh84yJXfytovILXOAQ	Golf, Active Life	Phoenix	NaN	0	33.5	-1
1	1	30 Eglinton Avenue W	{'RestaurantsReservations': 'True', 'GoodForMe...	QXAEGFB4oINsVuTFxEYKFQ	Specialty Food, Restaurants, Dim Sum, Imported...	Mississauga	{'Monday': '9:0-0:0', 'Tuesday': '9:0-0:0', 'W...	1	43.6	-7
2	2	10110 Johnston Rd, Ste 15	{'GoodForKids': 'True', 'NoiseLevel': "u'avera...	gnKjwL_1w79qoiV3IC_xQQ	Sushi Bars, Restaurants, Japanese	Charlotte	{'Monday': '17:30-21:30', 'Wednesday': '17:30-...	1	35.1	-8
3	3	15655 W Roosevelt St, Ste 237	NaN	xvX2CttrVhyG2z1dFg_0xw	Insurance, Financial Services	Goodyear	{'Monday': '8:0-17:0', 'Tuesday': '8:0-17:0', ...	1	33.5	-1

	Unnamed: 0	address	attributes	business_id	categories	city	hours	is_open	latitude	lo
4	4	4209 Stuart Andrew Blvd, Ste F	{'BusinessAcceptsBitcoin': 'False', 'ByAppoint...	HhyxOkGAM07SRYtlQ4wMFQ	Plumbing, Shopping, Local Services, Home Servi...	Charlotte	{'Monday': '7:0-23:0', 'Tuesday': '7:0-23:0', ...	1	35.2	-8

```
# drop all missing value for business data and reset index
business = business.dropna(axis=0, how='any', subset=['categories']).reset_index(drop=True)
```

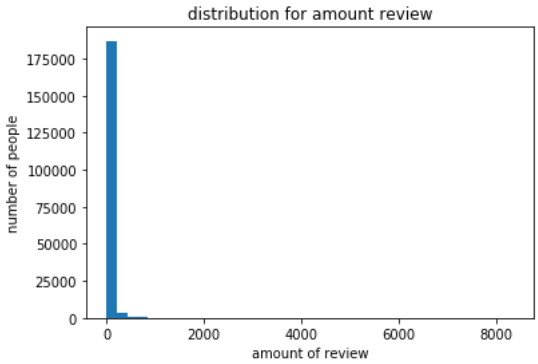
```
business = business.drop(columns=["address", "is_open", "postal_code"])
```

Considering the following hitogram and corresponding statistics for number of review received for all business, the histogram is unimodal. Amoung 192127 business, the average number of review received per business is around 33.6. The mimimum value of review recieved is 3.

```
pd.set_option("precision", 1)
business.review_count.describe()
```

```
count    192127.0
mean         33.6
std         110.3
min           3.0
25%           4.0
50%           9.0
75%          25.0
max        8348.0
Name: review_count, dtype: float64
```

```
fig = plt.figure()
plt.hist(business.review_count, bins=40)
plt.title("distribution for amount review")
plt.xlabel("amount of review")
plt.ylabel("number of people")
plt.show()
```



Overview of all business provided by Yelp

Before starting to analysis, let's dig into the dataset.

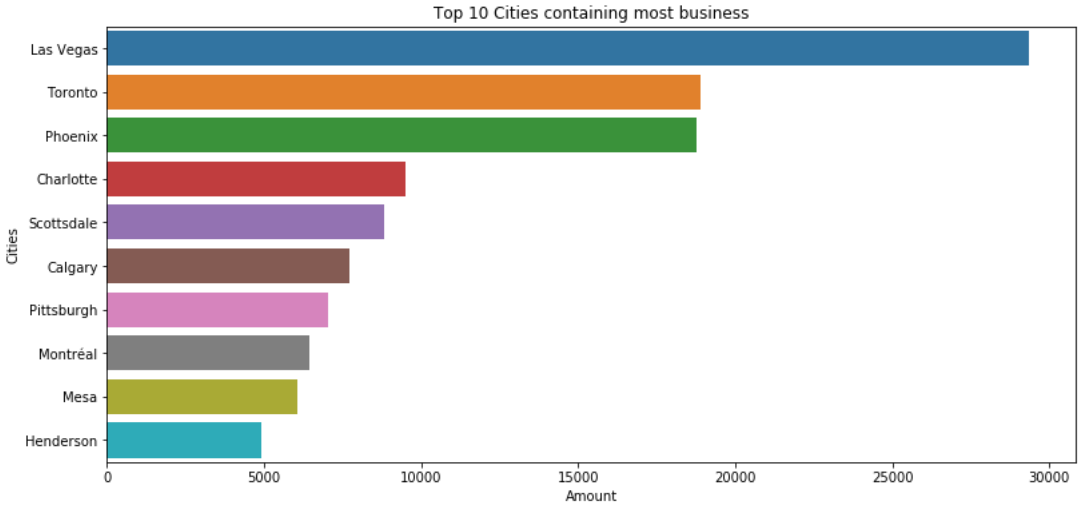
The dataset provides information about business form 1204 cities. I use bar plot to show the top 10 cities contain most of business. Top 1 city is Las Vegas, where the information 27000 business could be found in yelp, followed by Toronto, the number of business is around 17000. You might notice, the top 10 country are from North American, which is a surprisig result. indicating yelp primary target are people from North American, is it really the cases ?

```
# create data frame contains all cities with number of business per cities
business_to_city = pd.DataFrame(business.city.value_counts()).rename(columns={"city" : "Amount business"})
```

```
business_to_city.count()[0]
```

```
1201
```

```
fig = plt.figure(figsize=(13,6))
sns.barplot(business.city.value_counts().values[:10], business.city.value_counts().index[:10])
plt.title("Top 10 Cities containing most business")
plt.xlabel("Amount")
plt.ylabel("Cities")
plt.show()
```



Are yelp primary target are people from North American ?

Since the data only contains State information, it is hard to visualize from the graph. Therefore, I extracted the country name and corresponding countries' location based on their city name, and plot the number of businesses per country using bar plot and map.

The countries's name could be updated easily using geolocator function from Geopy API (https://geopy.readthedocs.io/en/stable/). Geolocator takes city name as argument, and return the country name and corresponding address. Since we only want the distribution of number of business across regions, it is too time consuming and unnecessary to extract the information for those 1200 cities in the data set. Therefore, for each state in the data, I only pick one city belonged to the state, and use city to get the longtitude and latitude.

```
df_city = business.groupby(["city", "state"], as_index=False).count()[["city", "state", "categories"]]\
.sort_values('categories', ascending=False).rename(columns = {"categories": "amount"})
df_city.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	city	state	amount
482	Las Vegas	NV	29285
1115	Toronto	ON	18874
820	Phoenix	AZ	18697
172	Charlotte	NC	9487
1004	Scottsdale	AZ	8824

```
state_location = {}
# get all state
```

```
all_state = df_city.state.unique()
# use geolocator api
geolocator = Nominatim(user_agent="my application")

for state in all_state:
    # pick the first city from each state
    city = business[business.state == state].city.values[0]
    # get location, and use location to get the lon and lat
    location = geolocator.geocode(city)
    if(location != None):
        state_location[state] = [location.address.split(",")[-1].strip(), location.longitude, location.latitude]
    else:
        state_location[state] == "None"
```

```
def get_country(row, column):
    if column == "country":
        return state_location[row["state"]][0]
    elif column == "lon":
        return state_location[row["state"]][1]
    else:
        return state_location[row["state"]][2]
```

```
# add the country, lon, lat into df_city dataframe
df_city["country"] = df_city.apply(lambda row: get_country(row, "country"), axis=1)
df_city["lon"] = df_city.apply(lambda row: get_country(row, "lon"), axis=1)
df_city["lat"] = df_city.apply(lambda row: get_country(row, "lat"), axis=1)
```

```
df_city.head()
```

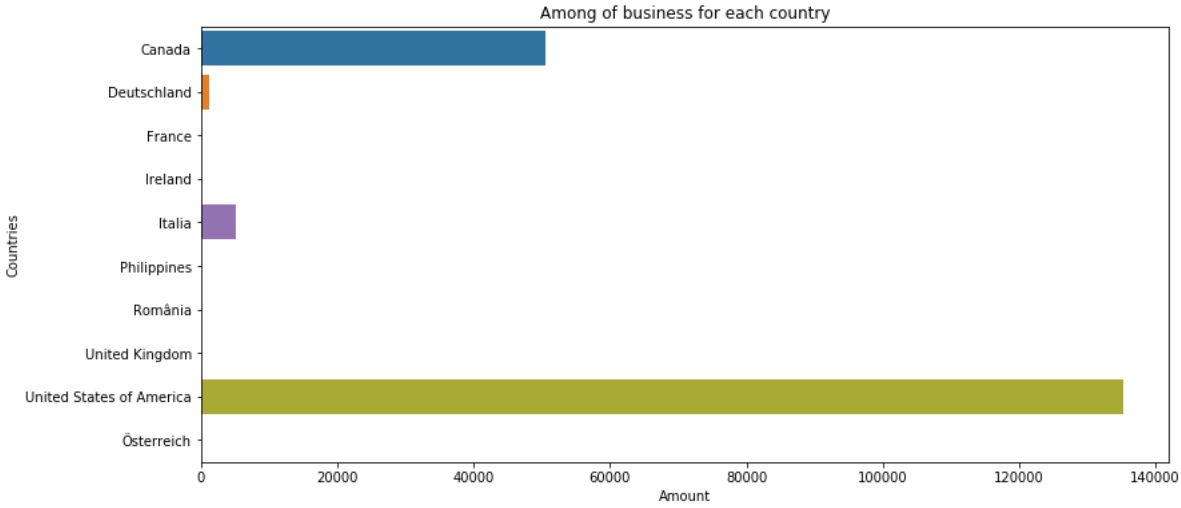
```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	city	state	amount	country	lon	lat
482	Las Vegas	NV	29285	United States of America	-115.148516	36.167256
1115	Toronto	ON	18874	Canada	-79.645729	43.590338
820	Phoenix	AZ	18697	United States of America	-112.074142	33.448437
172	Charlotte	NC	9487	United States of America	-80.843127	35.227087
1004	Scottsdale	AZ	8824	United States of America	-112.074142	33.448437

```
# get amount of business for country
country_amount = df_city.groupby(["country"], as_index=False).sum()[["country","amount"]]
```

The following bar plot shows the distribution of business for each country. We might notice that the data set only contains following 10 countries. Most businesses are from America, followed by Canada, and Italia, which verifies my assumption that yelp primary customer are people from North America. The map also suggests the same finding, therefore, the finding of this data analysis might not be suitable for people outside North America. Then, I will see all category of business in Yelp to get a deeper understanding on yelp data set.

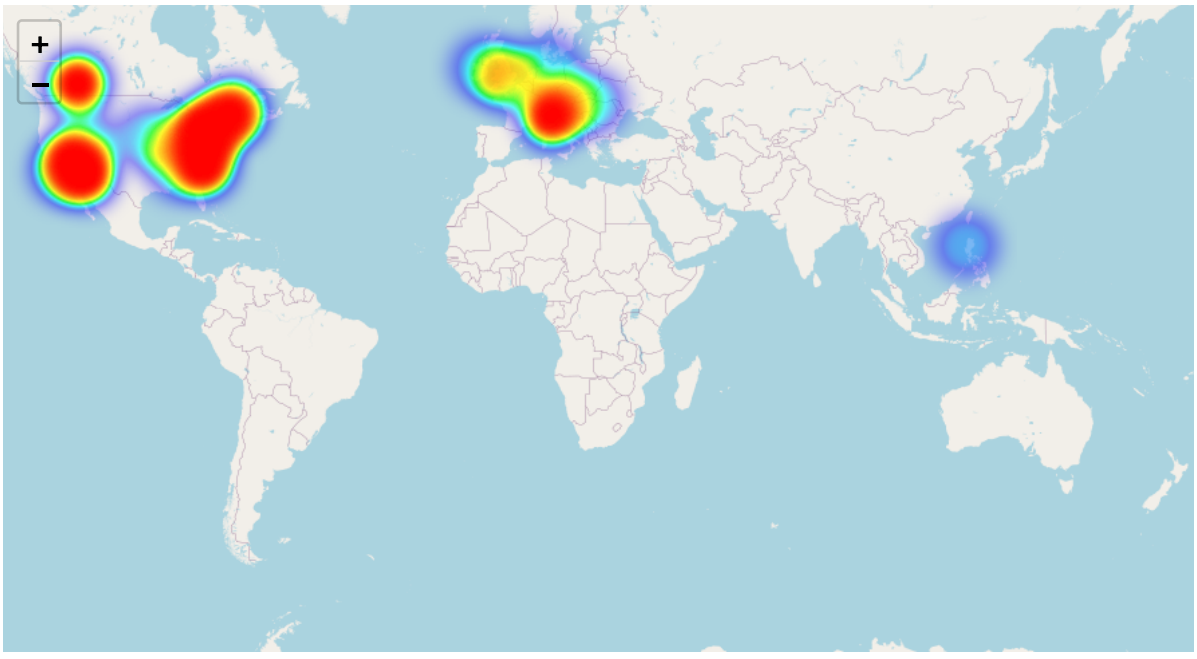
```
fig = plt.figure(figsize=(13,6))
sns.barplot(country_amount["amount"], country_amount["country"])
plt.title("Among of business for each country")
plt.xlabel("Amount")
plt.ylabel("Countries")
plt.show()
```




```
location_amount = df_city.groupby(["lon", "lat"], as_index=False).sum()[["lon","lat","amount"]]
```

```
amount, lon, lat = [], [], []
for i, row in location_amount.iterrows():
    amount.append(row["amount"])
    lon.append(row["lon"])
    lat.append(row["lat"])
```

```
data = [[lat[i],lon[i],amount[i]] for i in range(len(amount))]
world_map = folium.Map(location=[0,0], zoom_start=2)

HeatMap(data).add_to(world_map)
world_map
```



What are the most frequent business categories?

You might notice each business might belong to multiple category, while each categories are seperated by comma. Using python split function could help us extract all categories for each business. There are 1300 different categories in totall, since we only interested in most frequent business categories, We come to distribution the top

10 categories.

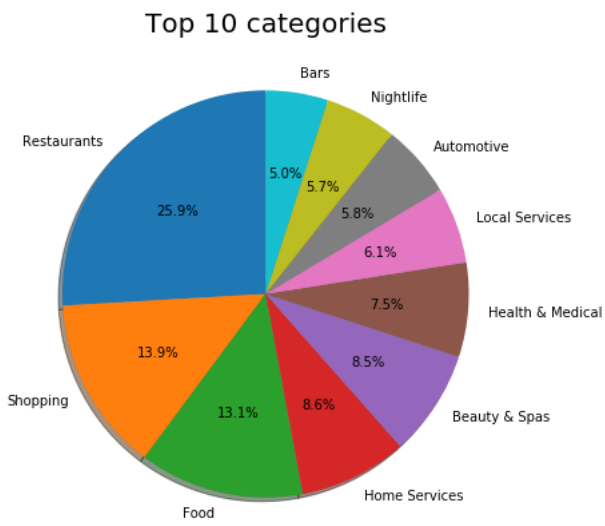
It is surprsing that Yelp is not a review platform containing only restaurants. Following graph shows the top 10 category among all businesses. Unsurprisingly, restuarants are most popular categories in yelp, around 26% bussiness among 10 categories are restaurants, while foods counts for 13%. Considering one business could have multiple categories, the percentage of restuarants might be overestimated since the category for restuarants tend to include food. The second popular categories for all bussiness are shopping, counting for 14%, which suggests that most of people use yelp for dinning and shoppin. Since this data analysis mainly focus on restaurants, I decide to look deeper to dicide which food and which types of restaurants people tend to go.

```
# get all category based on given dataframe
def top_categories(df, top):

    # extra all categories, each element is a string of multiple categories separated by comma
    all_category = df.categories
    # key is category, values is the number of business corresponding to its category.
    category_to_number = {}

    for categories in all_category:
        temp = categories.split(',')
        for category in temp:
            category_to_number[category.strip()] = category_to_number.get(category.strip(), 0) + 1
    # get top category
    top_10_category = pd.Series(category_to_number).sort_values(ascending=False)[:top]
    return top_10_category, len(category_to_number.keys())
```

```
# get top 10 category and overall amount of categories
top_10_category, category_amount = top_categories(business, 10)
print("The number of category is " + str(category_amount))
```

```
The number of category is 1300
```

```
fig, ax = plt.subplots(figsize=(7, 7))
ax.pie(top_10_category,  labels=top_10_category.index, autopct='%1.1f%%', shadow=True, startangle=90)
plt.title("Top 10 categories", fontsize=20)
plt.show()
```



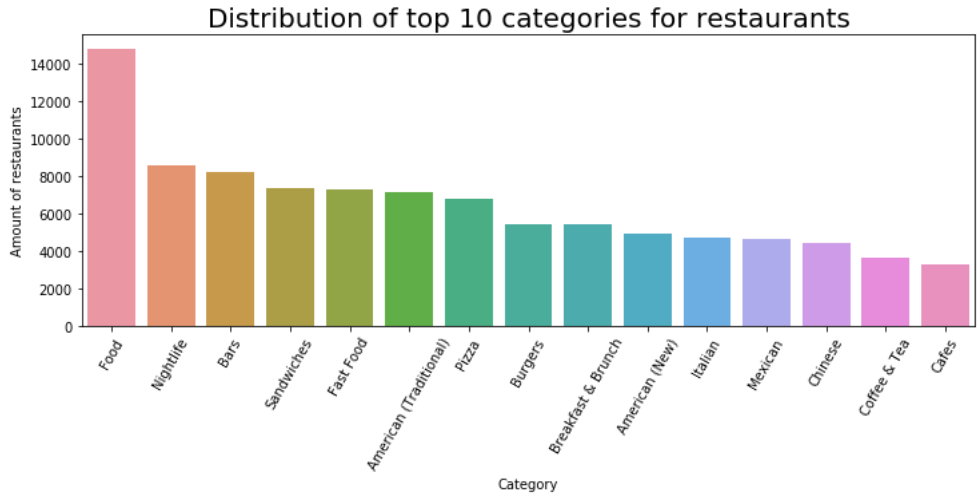
Which types of restuarant are more popular ?

After finding popular categories, we come to look at popular categories for restaurants. Observing following bar plot, Aftrt excluding category "Food", the top 1 and top 2 categories are nightlife and bar, indicating people in North America tend to go to restuarant enjoying night life, drinking with friends, while for food categories, most restaurants are fast food restaurants serving sandwiches, pizza and burgers. This makes sense, considering the popularity of fast food in North American especially for US. Meanwhile, Italian, mexican and chinese restaurants are more popular compared with other cuisines.

```
# get all restaurants
restaurants = business[business.categories.str.contains("Restaurants")]
```

```
# since the first category is restaurant, we ignore it
restaurants_category = top_categories(restaurants, 16)[0][1:]
```

```
fig = plt.figure(figsize=(12,4))
ax = sns.barplot(restaurants_category.index, restaurants_category.values,)
plt.title("Distribution of top 10 categories for restaurants", fontsize=20)
plt.xticks(rotation=60)
plt.ylabel('Amount of restaurants')
plt.xlabel('Category')
plt.show()
```



Which categories tend to have Bike parking ?

For most of customer ride bikes outside, the biggest problem to consider is where can I parking. To answer this questions, let's first consider the the categories contains highest numbero of parking.

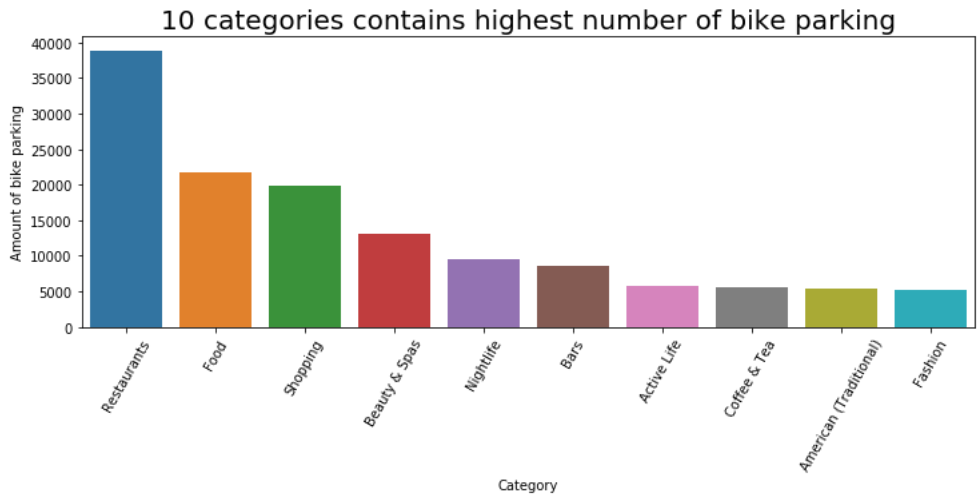
Firstly, I select all data with bike parking based on attributes in yelp dataframe. Th following bar plot shows the distirbution of highest number of bike parking for top 10 categories. Restaurants are top 1 category to have bie parking, there are more than 35000 restaurant having bike parkings, followed by Food and Shopping.

```
# remove missing value in attributes
df_parking = business[~business.attributes.isna()]
```

```
category_to_parking = {}
for i, row in df_parking.iterrows():
    if "BikeParking" in row["attributes"]:
        temp = row.categories.split(',')
        for category in temp:
            category_to_parking[category.strip()] = category_to_parking.get(category.strip(), 0) + 1
```

```
top_10_category_bikepaking = pd.Series(category_to_parking).sort_values(ascending=False)[:10]
```

```
fig = plt.figure(figsize=(12,4))
ax = sns.barplot(top_10_category_bikepaking.index, top_10_category_bikepaking.values,)
plt.title("10 categories contains highest number of bike parking", fontsize=20)
plt.xticks(rotation=60)
plt.ylabel('Amount of bike parking')
plt.xlabel('Category')
plt.show()
```



However, it is hard to draw conclusion baded on the distribution since the total number of business have a huge impact on the number of business who have bike parking. To eliminate the effect of number, I decide to calculate the ratio of business who has bike parking among all categories. Before staring to make anaylsis, I remove all category with business less than 10 since some business are unique and it is meaningless to consider the category only having small amount of business.

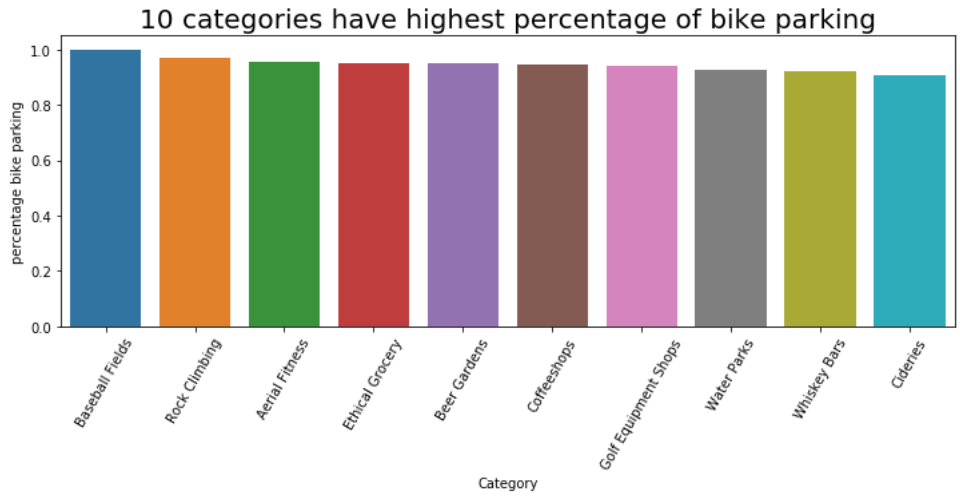
```
all_category = top_categories(business, -1)[0]
```

```
# calculate the ratio of business who has bike parking among all categories
category_parking_ratio = {}
parking_category = category_to_parking.keys()
for category in parking_category:
    # only select the category with business more than 10
    if category_to_parking[category] >= 10:
        category_parking_ratio[category] = round(category_to_parking[category] / all_category[category], 3)
```

```
# get top 10 categories
top_10_category_ratio = pd.Series(category_parking_ratio).sort_values(ascending=False)[:10]
```

The following bar plots shows top 10 categories with highest percentage of bike parking. The graph suggests that every baseball fields offer bike parking, followed by Rock climbing and Aerial Fitness, indicating sports related business tend to offer bike parkings.

```
fig = plt.figure(figsize=(12,4))
ax = sns.barplot(top_10_category_ratio.index, top_10_category_ratio.values)
plt.title("10 categories have highest percentage of bike parking", fontsize=20)
plt.xticks(rotation=60)
plt.ylabel('percentage bike parking')
plt.xlabel('Category')
plt.show()
```



Noticing only 65% restaurants offer bike parking, therefore next time you want ride bicycle to a restaurants a, check websites to see whether it offers bike parking first.

```
print("ratio for restaurant to have bike parking: " + str(category_parking_ratio["Restaurants"]))
```

```
ratio for restaurant to have bike parking: 0.655
```

Does more yelp reviews lead to a higher rating ?

The number of review received by local business does reflect the popularity of the business, since only small portion of customer provides review for those businesses. Some people recently claims more review could lead to higher rating, and increase business sales. Is it really the cases?

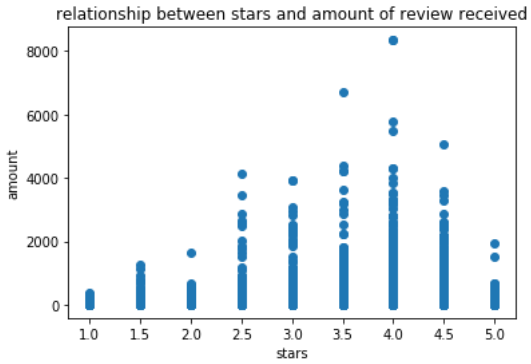
To answer the questions, I first plot the the scatter point showing the relationship between review count and stars, there is a pretty week relationship between stars and review amount, since most of business with review amount higher than 2000 have stars higher than 2.5

To exam deeper, I plot the following two graph, the first plot showing the the total amount of review for each stars, while the second plot shows the distribution of average amount of review. Both of two graph shows similar trends that business with stars 4.0 tend to have more reviews. Considering 4.0 is actually a pretty high score, higher number of review seem to lead to a high score.

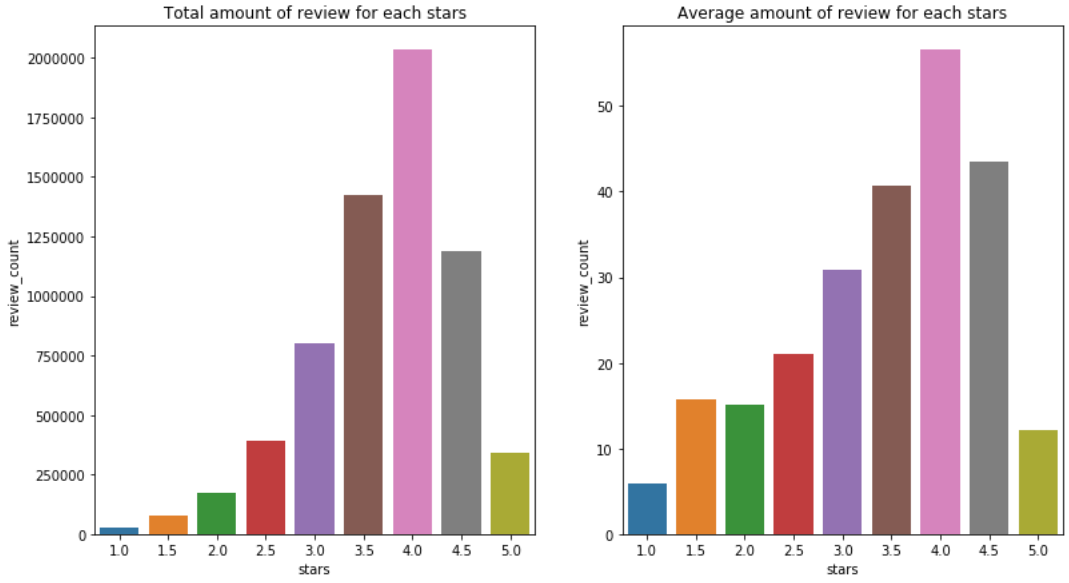
However, we need to consider the fact that the type of business, the quarlity of service and the quality of the food have a huge impact on people's perference, to better eliminate those effect, I decide to plot the distribution of stars and review received by Mac Donald's and KFC. Since those two restaurants offer standardized food and services.

The finding is not surprising, most of people tends to give 1.0 when eating at those two restarurants, there is no evidentce indicating more reviews could lead to higher rating.

```
fig = plt.figure()
plt.scatter(business.stars, business.review_count)
plt.title("relationship between stars and amount of review received")
plt.xlabel("stars")
plt.ylabel("amount")
plt.show()
```



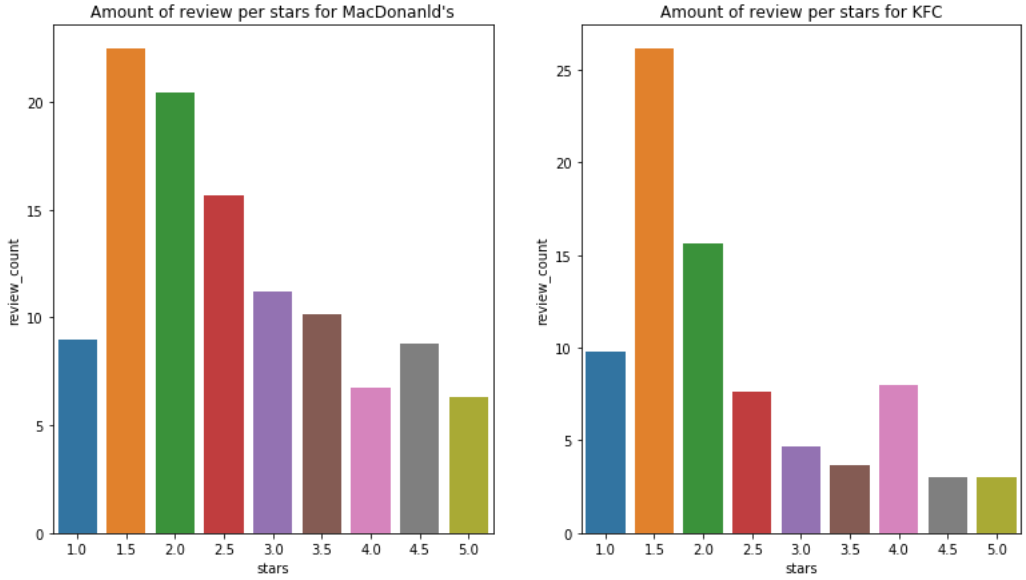

```
# get sum of review for each stars
sum_review = business.groupby(["stars"], as_index=False)[["review_count"]].sum()
# get average number of review for each stars
average_review = business.groupby(["stars"], as_index=False)[["review_count"]].mean()

fig, ax= plt.subplots(1, 2, figsize=(13, 7))
sns.barplot(x='stars', y="review_count", data=sum_review,ax=ax[0])
sns.barplot(x='stars', y="review_count", data=average_review, ax=ax[1])
ax[0].set_title("Total amount of review for each stars")
ax[1].set_title("Average amount of review for each stars")
plt.show()
```




```
mcDonald = business[business["name"].str.contains("McDonald")]
kfc = business[business["name"].str.contains("KFC")]
```

```
macDonanld_review = mcDonald.groupby(["stars"], as_index=False)[["review_count"]].mean()
kfc_review = kfc.groupby(["stars"], as_index=False)[["review_count"]].mean()
```

```
fig, ax= plt.subplots(1, 2, figsize=(13, 7))
sns.barplot(x='stars', y="review_count", data=macDonanld_review, ax=ax[0])
ax[0].set_title("Amount of review per stars for MacDonanld's")
ax[1].set_title("Amount of review per stars for KFC")
sns.barplot(x='stars', y="review_count", data=kfc_review, ax=ax[1])
plt.show()
```



To confirm my finding, I calculate correlation between two variables, the correlation is 0.04 indicating there is no linear relationship between stars and number of review .

```
cov_array = np.cov(business.stars, business.review_count)
# calculate the covariance between trip_duration_seconds and dis tance
covariance = cov_array[0][1]
# correlation = covariance/(std(X) * std(Y))
correlation = covariance /( math.sqrt(cov_array[0][0]) * math.sqrt(cov_array[1][1]))
print("correlation is: " +  str(correlation))
```

```
correlation is: 0.04003808039288897
```

After having a brief understanding about Yelp data set, Let's foucs on All business in Toronto. Toronto, as previouly seen, is the second biggest cities in Yelp containing 18874 business. Let's look at the all categories in GTA, and compare the result with rest of the world

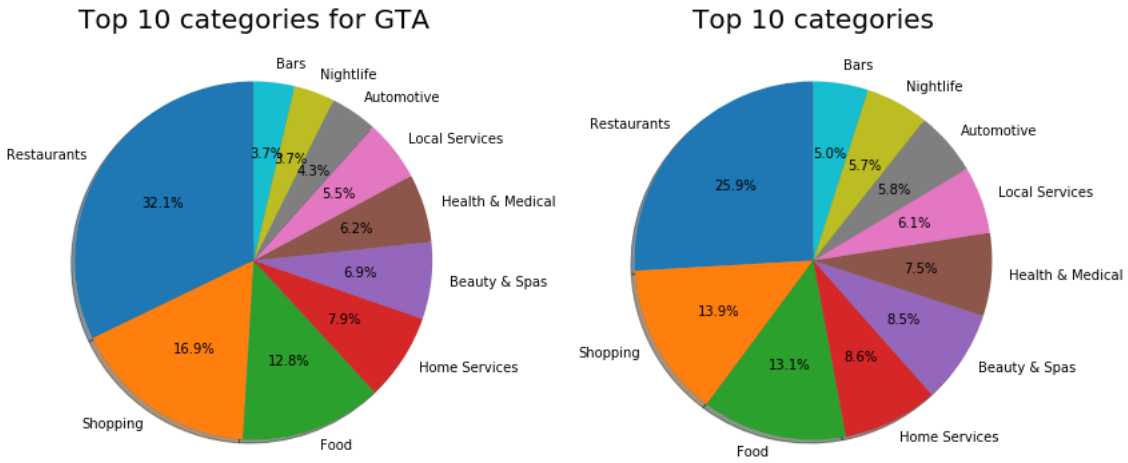
```
# get all data in GTA region
GTA_business = business[business.city == "Toronto"]
```

What are the most frequent business categories in GTA ?

The following graph shows the top 10 categories for GTA, and top 10 categories for whole world (mainly North American). It is unsurprising that two pie chart shows the same finding, the most popular category are restaurants, followed by shopping since the most yelp data are from North America. One implication is that, bar and nightlife is not important for people in GTA compared with people in other regions, the percentage bar in GTA is around 3.7% which is 1.3 % less than the percentage of bars in whole world.

```
# 10 top category in GTA
GTA_top_category, category_amount = top_categories(GTA_business, 10)
```

```
fig, ax= plt.subplots(1, 2, figsize=(12, 12))
ax[0].pie(GTA_top_category,  labels=top_10_category.index, autopct='%1.1f%%', shadow=True, startangle=90)
ax[1].pie(top_10_category,  labels=top_10_category.index, autopct='%1.1f%%', shadow=True, startangle=90)
ax[0].set_title("Top 10 categories for GTA", fontsize=20)
ax[1].set_title("Top 10 categories", fontsize=20)
fig.tight_layout()
plt.show()
```

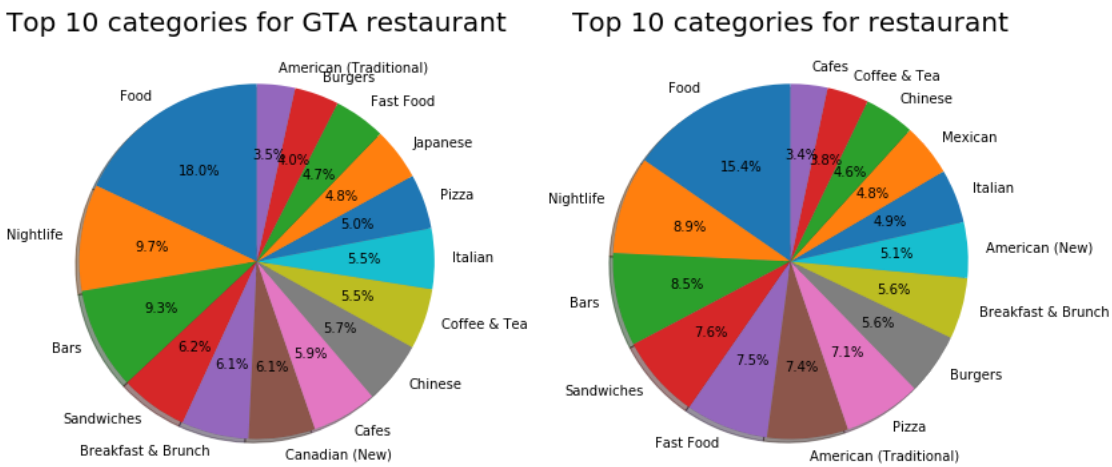


If we come to look at the restaurant in GTA compared with restaurant all over the world. Both of the pie plot reveal tha the categories for GTA restaurant are similar to categories for restaurant over the world. Canadian people also love fast food including piazza, burgers and sandwiches. One difference is that, people in toronto prefer Japanese cuisine, which counts for 4.8%. Meanwhile people in Toronrto are likely to eat canadian cuisine.

```
# get all restaurants in GTA
GTA_restaurants = GTA_business[GTA_business.categories.str.contains("Restaurants")]
```

```
GTA_restaurants_category = top_categories(GTA_restaurants, 16)[0][1:]
```

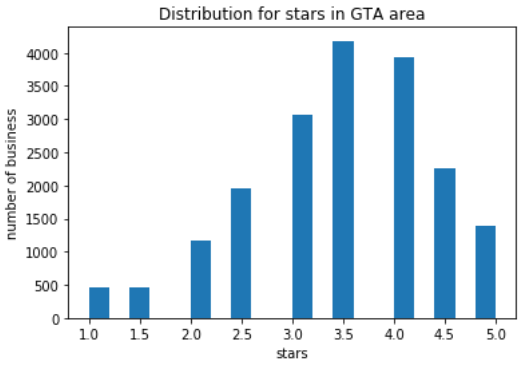
```
fig, ax= plt.subplots(1, 2, figsize=(12, 12))
ax[0].pie(GTA_restaurants_category,  labels=GTA_restaurants_category.index, autopct='%1.1f%%', shadow=True,
startangle=90)
ax[1].pie(restaurants_category,  labels=restaurants_category.index, autopct='%1.1f%%', shadow=True, startangle=90)
ax[0].set_title("Top 10 categories for GTA restaurant", fontsize=20)
ax[1].set_title("Top 10 categories for restaurant", fontsize=20)
fig.tight_layout()
plt.show()
```



What are best business in Toronto ?

The following graph shows disturbution of stars for business located in GTA are. The histogram is unimodal showing the average of stars for each business are 3.5 to 4, indicating people in Toronto tends to give a positive review. However, what is the best business in Toronto ?

```
fig = plt.figure()
plt.hist(GTA_business.stars, bins = 20)
plt.title("Distribution for stars in GTA area")
plt.xlabel("stars")
plt.ylabel("number of business")
plt.show()
```



Before staring doing data analysis, a common question is what is best business mean? How we should define it. I personally define best franchises into following two ways:

```
- business with highest popularity
- business with highest stars
```

Therefore, we want to find the business with highest popularity and highest stars. Firstly, I fiter the top 0.1% business by selecting all business with number of review received higher than 99.9% quantile, since review amount is an important factor when considering business popularity. Then, I apply the same methods on remaining business, and removing all business with stars less than 99.9 quantile. There are 4 businesses remaining.

```
- Byblos
- Banh Mi Boys
- Seven Lives Tacos Y Mariscos
- Pai Northern Thai Kitchen
```

The top one business are Pai Northern Thai Kitchen which receives 2121 reviews and average stars are 4.5 ! Great Job !

```
# cutoff for stars 99.9 quantile
cutoff_amount_review =  GTA_business["review_count"].quantile(0.999)
# among business with highest review, we choose business with highest stars
business_with_high_amount_review = GTA_business[GTA_business.review_count > cutoff_amount_review]
cutoff_stars = business_with_high_amount_review["stars"].quantile(0.999)
```

```
best_business = business_with_high_amount_review[business_with_high_amount_review.stars >= cutoff_stars]
```

```
best_business.reset_index()[["name", "review_count", "stars", ""]]
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	name	review_count	stars
0	Byblos	700	4.5
1	Banh Mi Boys	1045	4.5
2	Seven Lives Tacos Y Mariscos	1152	4.5
3	Pai Northern Thai Kitchen	2121	4.5

Does restaurants location play an important role in reviews?

Business location is a important factor to consider when opening a new business, a good business location mighthelp local business attract more customers. I am curious on the impact of location on business's popularity. Since restuarants are most popular categories among all yelp business and restiarants are closely related to our daily life, I decide to focus on restaurants.

I previously define popularity as the number of review recieved by each restaurants. To answer the question, we need to find two groups of restaurant, popular restaurants and unpopular restaurants. Therefore, to extract all popular restaurant in GTA, I select all restaurants with amount of reviews received more than 99.5 percent qunatile and I define unpopular restaurants as the restaurants with amount of reviews recived less than 0.05 % qunatile.

```
# select the most popular restarurant who has number of review greater than 99.5% qunatile
high_cutoff =  GTA_restaurants["review_count"].quantile(0.995)
popular_business = GTA_restaurants[GTA_restaurants.review_count > high_cutoff]
```

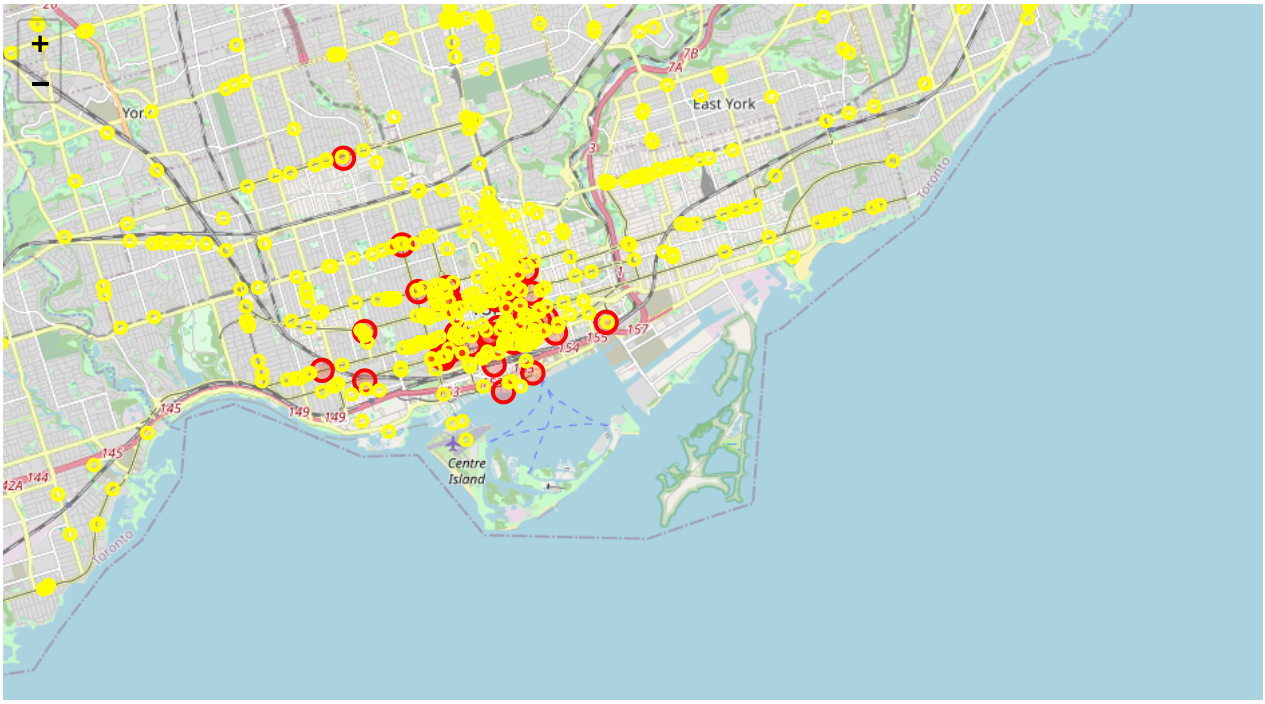
```
# select the least popular restarurant who has number of review less than 0.05% qunatile
low_cutoff =  GTA_restaurants["review_count"].quantile(0.005)
unpopular_business = GTA_restaurants[GTA_business.review_count <= low_cutoff]
```

The following map shows the locations of popular restaurants and unpopular restaurants, red dots indicate popular restaurants, while yellow dots indicate the unpopular one. Notice the amount of unpopular restaurants is significant higher than the amount of popular restaurants since most restaurants only contain 3 reviews.

From the map, we could observes the all unpopular restaurants are distributed randomly on the map, indicating no relationship between location and popularity. However, if we look at all popular restaurants, most of them are displayed within downtown area of Toronto near the Lake , only few popular restaurants are outside downtown, indicating the restaurants located in downtown have higher chance to become popular.

```
# reference https://www.zhihu.com/question/33783546/answer/775946401

latitude = 43.653225
longitude = -79.383186

#create toronto map
toronto_map = folium.Map(location=[latitude, longitude], zoom_start=12)

incidents = folium.map.FeatureGroup()

# display all popular business

for i, row in popular_business.iterrows():
    incidents.add_child(
        folium.CircleMarker(
            [row["latitude"], row["longitude"]],
            radius=8, # define how big you want the circle markers to be
            color='red',
            fill=True
        )
    )
```

```
# display all unpopular business
for i, row in unpopular_business.iterrows():
    incidents.add_child(
        folium.CircleMarker(
            [row["latitude"], row["longitude"]],
            radius=4, # define how big you want the circle markers to be
            color='yellow',
            fill=True
        )
    )


# Add incidents to map
toronto_map = folium.Map(location=[latitude, longitude], zoom_start=12)
toronto_map.add_child(incidents)
```



Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

However, the appraoch exists limitation. Firstly, we cannot guarantee yelp contains all popular business in Toronto, since Yelp only contains a small portion of business in Toronrto. Secondly, the yelp primary user's are certain groups of people, not everyone will use yelp, for me. I never use yelp for reviewing purposes. The conclusion drawn by yelp data might be biased.

Coffee is more than just a hot dark delicious drink, it is a way of life. From previous pie chart about all categories in Toronrto, we might notice 6 % restaurants in Toronto are cafe, indicates cafe is extreme popular in people in GTA area.

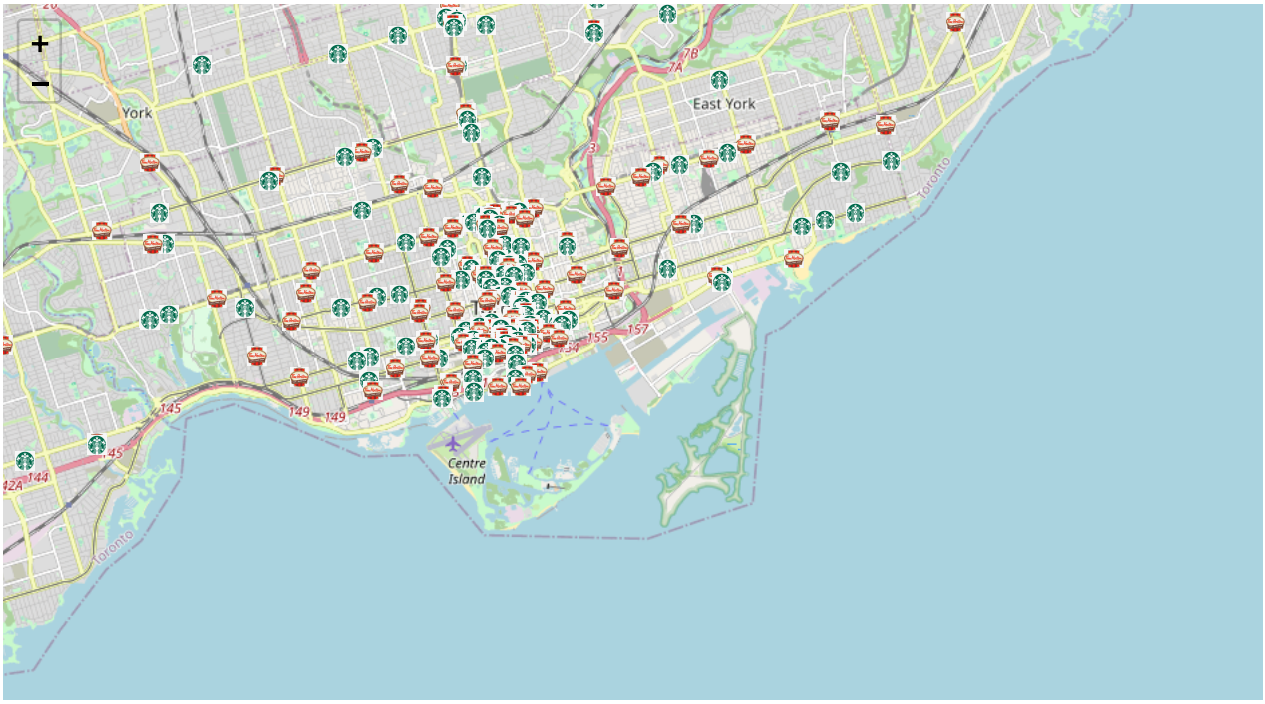
Starbucks and Tim horton's, two world most successful coffee company, both of them are well-known by thier coffee. You could find them everywhere in GTA. Only look at GTA area, there are 156 starbucks and 124 Tim Hortons. I distributes all starbucks and Tim Horton's in the following map. Notice most of Starbucks and Tim Horton's are concentrated in Toronto downtown area, each cafe are close with each others. Let's make further analysis on the distribution for starbucks and Tim Horton's

```
starbucks = GTA_business[GTA_business["name"].str.contains("Starbucks")].reset_index()
starbucks["name"] = "Starbucks"
tim = GTA_business[GTA_business["name"].str.contains("Tim Horton")].reset_index()
tim["name"] = "Tim Horton"
coffee = pd.concat([starbucks, tim])
```

```
print("the number of starbucks in GTA is: " + str(len(starbucks)))
print("the number of Tim Horton's in GTA is :" + str(len(tim)))
```

```
the number of starbucks in GTA is: 156
the number of Tim Horton's in GTA is :124
```

```
latitude = 43.653225
longitude = -79.383186

toronto_map = folium.Map(location=[latitude, longitude], zoom_start=12)

for i, row in coffee.iterrows():

    if row['name'] == 'Starbucks':
        icon = folium.features.CustomIcon('starbucks.png' ,icon_size=(14, 14))
        pop_up = 'Starbucks'
    else:
        icon = folium.features.CustomIcon('tim.png' ,icon_size=(14, 14))
        pop_up = "Tim Horton's"

    marker = folium.Marker(
        [row.latitude, row.longitude],
        icon=icon,
        popup=pop_up
    )

    toronto_map.add_child(marker)

toronto_map
```



Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

Is it true that for every Tim Hortons in the GTA there is a Starbucks nearby ?

To answer this questions, I divides starbucks into following three categories based on the range of distance between Tim horton's and Starbucks -Starbucks having Tim Horton's with 200 meters -Starbucks having Tim Horton's with 500 meters -Starbucks having Tim Horton's with 1000 meters Firstly, for starbucks dataframe, I add three columns: 200m, 500m, 1000m to show whether there is a Tim horton's within certain ranges. For each starbucks in the dataframe, I calculate the distance bewteen this starbuck and Tim horton's in Tim horton's dataframe, and assign the result based on three categories.

```
# if there is a tim horton within dis distance, return True
def is_tim_near_by(row1, df, dis):
    for i, row2 in df.iterrows():
        if (distance.distance((row1["latitude"], row1['longitude']), (row2["latitude"], row2['longitude'])).m < dis):
            return "True"
    return "False"
```

```
starbucks["200m"] = starbucks.apply(lambda row: is_tim_near_by(row, tim, 200), axis=1)
starbucks["500m"] = starbucks.apply(lambda row: is_tim_near_by(row, tim, 500), axis=1)
starbucks["1000m"] = starbucks.apply(lambda row: is_tim_near_by(row, tim, 1000), axis=1)
```

```
starbucks.head()
```

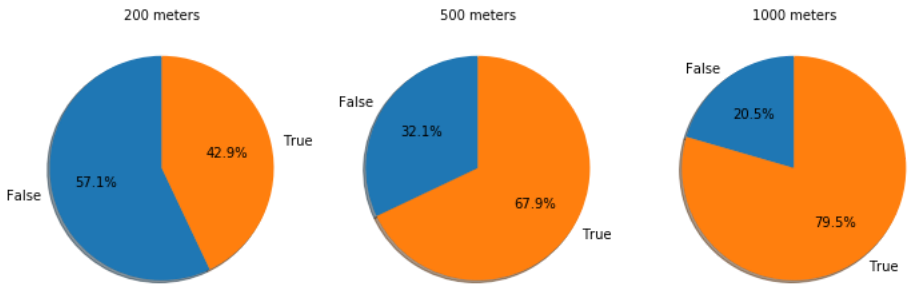
```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	index	Unnamed: 0	attributes	business_id	categories	city	hours	latitude	longitude	name
0	2295	2300	{'BikeParking': 'True', 'BusinessParking': "{'...	2iYGfGTI2Bm56rUaJfpZug	Coffee & Tea, Food	Toronto	NaN	43.7	-79.3	Starbucks
1	2589	2595	{'BikeParking': 'True', 'BusinessParking': "{'...	PNzGMlH0SnvIMxIESVNLgw	Food, Coffee & Tea	Toronto	NaN	43.6	-79.4	Starbucks
2	3821	3834	{'RestaurantsDelivery': 'False', 'RestaurantsT...	POuD26KrQ_ssxKuSFfXY0A	Food, Bakeries, Coffee & Tea	Toronto	{'Monday': '7:30-17:0', 'Tuesday': '7:30-17:0'...	43.7	-79.6	Starbucks
3	4641	4656	{'BusinessParking': " {'garage': False, 'street...	uWbmDMJcuxVgryNihVx3rw	Cafes, Food, Coffee & Tea, Restaurants	Toronto	{'Monday': '6:0-23:0', 'Tuesday': '6:0-23:0', ...	43.7	-79.4	Starbucks
4	8275	8296	{'RestaurantsPriceRange2': '2', 'BusinessParki...	dZWh_lwv3EhZU-bKMiSQWA	Coffee & Tea, Food	Toronto	{'Monday': '6:0-22:0', 'Tuesday': '6:0-22:0', ...	43.7	-79.3	Starbucks

```
# get retio for distance 200m, 500m, 1000m for plotting
twohundred = starbucks.groupby("200m", as_index=False).count()[["200m", "index"]]
fivehundred = starbucks.groupby("500m", as_index=False).count()[["500m", "index"]]
onethousand = starbucks.groupby("1000m", as_index=False).count()[["1000m", "index"]]
```

The following pie charts indicates 42.9% starbucks having Tim horton's located within 200 meters, if we increases the distance to 500 meters, we find 68% starbucks having Tim horton's nearby. Around 80% starbucks, you could find a Tim horton's within 1 kilometers.

```
fig, ax= plt.subplots(1, 3, figsize=(10, 10))
ax[0].pie(twohundred["index"],  labels=twohundred["200m"], autopct='%1.1f%%', shadow=True, startangle=90)
ax[1].pie(fivehundred["index"],  labels=fivehundred["500m"], autopct='%1.1f%%', shadow=True, startangle=90)
ax[2].pie(onethousand["index"],  labels=onethousand["1000m"], autopct='%1.1f%%', shadow=True, startangle=90)
ax[0].set_title("200 meters", fontsize=10)
ax[1].set_title("500 meters", fontsize=10)
ax[2].set_title("1000 meters", fontsize=10)
fig.tight_layout()
plt.show()
```



Then, I calculate the average number Tim Horton's nearby among those three categories and use the informatino to display the bar plot. Noticing from the bar plot, the average number of Tim horton's within 1500m from starbucks are 14, indicating for each starbucks, you could find 14 Tim Horton's within 1.5 km, for 1 km and 500 meters, the average number of Tim Horton's nearby are greater than 2.

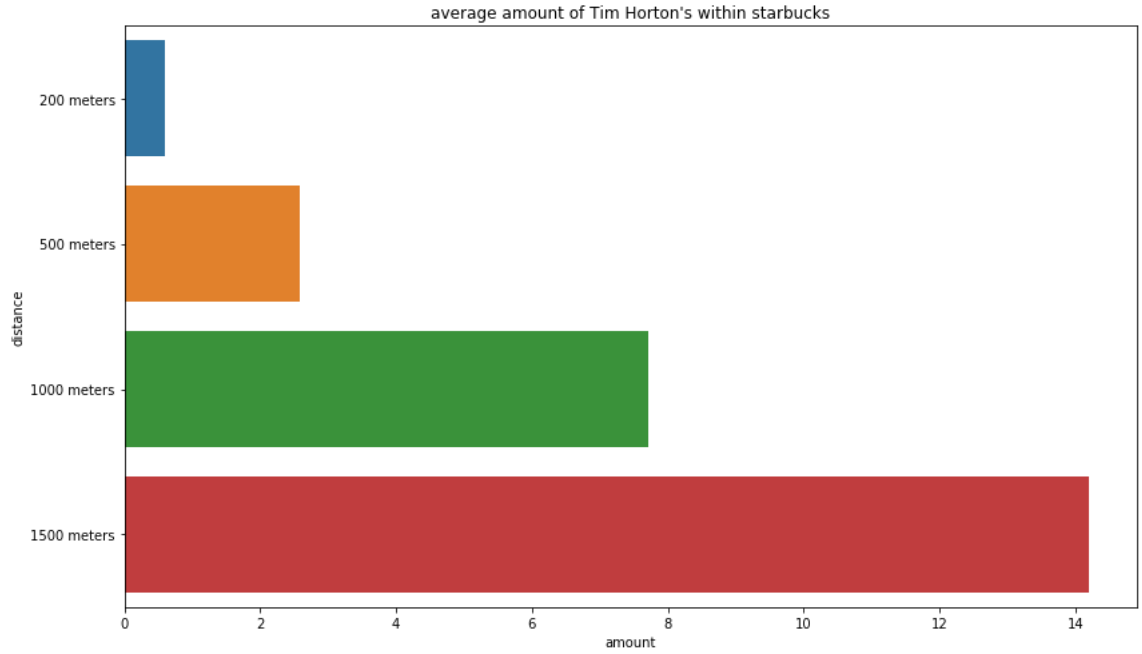
The finding suggests Starbucks and Tim horton's are usually close with each others.

```
# get the number of Tim Horton's with certain distance from starbucks
tim_within_starbuck = {"200 meters": 0 , "500 meters": 0, "1000 meters": 0, "1500 meters": 0}

for i, row1 in starbucks.iterrows():
    for j, row2 in tim.iterrows():
        # get distance and store it into tim_with_starbuck
        dst = distance.distance((row1["latitude"], row1['longitude']), (row2["latitude"], row2['longitude'])).m
        if dst < 1500:
            tim_within_starbuck["1500 meters"] += 1
            if dst < 1000:
                tim_within_starbuck["1000 meters"] += 1
                if dst < 500:
                    tim_within_starbuck["500 meters"] += 1
                    if dst < 200:
                        tim_within_starbuck["200 meters"] += 1
```

```
total_starbucks = len(starbucks)
```

```
# calculate average number of tim horton for each starbucks within given distance
for key, number in tim_within_starbuck.items():
    tim_within_starbuck[key] = number / total_starbucks
```

```
fig, ax= plt.subplots(figsize=(12, 7))
sns.barplot(list(tim_within_starbuck.values()), list(tim_within_starbuck.keys()), ax=ax)
plt.title("average amount of Tim Horton's within starbucks")
plt.xlabel("amount")
plt.ylabel("distance")
fig.tight_layout()
plt.show()
```



reference: https://nbviewer.jupyter.org/github/python-visualization/folium/tree/master/examples/

Then, it is time to make analysis on review of each business. The review data is a json file containing the informations about 6685900 reviews for all business in Yelp. Attribues of review table are as following: - review_id: ID of the review - user_id: ID of the user - business_id: ID of the business - stars: ratings of the business - date: review date - text: review from the user - useful: number of users who vote a review as usefull - funny: number of users who vote a review as funny - cool: number of users who vote a review as cool

Let's first loading the data.

```
# loading review data
temp = []
with open("yelp_dataset/review.json") as file:
    for line in file:
        temp.append(json.loads(line))
review = pd.DataFrame(temp)
```

```
review.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	business_id	cool	date	funny	review_id	stars	text	useful	user_id
0	ujmEBvifdJM6h6RLv4wQIg	0	2013-05-07 04:34:36	1	Q1sbwvVQXV2734tPgoKj4Q	1.0	Total bill for this horrible service? Over $8G...	6	hG7b0MtEbXx5QzbzE6C_VA
1	NZnhc2sEQy3RmzKTZnqtwQ	0	2017-01-14 21:30:33	0	GJXCdrto3ASJOqKeVWPi6Q	5.0	I *adore* Travis at the Hard Rock's new Kelly ...	0	yXQM5uF2jS6es16SJzNHfg
2	WTqjgwHlXbSFevF32_DJVw	0	2016-11-09 20:09:03	0	2TzJjDVDEuAW6MR5Vuc1ug	5.0	I have to say that this office really has it t...	3	n6-Gk65cPZL6Uz8qRm3NYw

	business_id	cool	date	funny	review_id	stars	text	useful	user_id
3	ikCg8xy5JIg_NGPx-MSIDA	0	2018-01-09 20:56:38	0	yi0R0Ugj_xUx_Nek0-_Qig	5.0	Went in for a lunch. Steak sandwich was delici...	0	dacAIZ6fTM6mqwW5uxkskg
4	b1b1eb3uo-w561D0ZfCEiQ	0	2018-01-30 23:07:38	0	11a8sVPMUFtaC7_ABRkmtw	1.0	Today was my second out of three sessions I ha...	7	ssoyf2_x0EQMed6fgHeMyQ

Is there a small group of users responsible for most reviews

It is necessary to figure out the user review pattern, since it could provide implication on how users make reviews. To prepare for data analysis, I groupby the the data based on user id and get the amount of review received for each user id.

The following data frame shows the top 5 users and corresponding reviews amount. The top 1 user provides 4129 reviews, followed by 2354, notice that the pattern is pretty strange.

The I sum the amount of reviews provided by top 5% users, 10% users and 20% users and distributes following pie chart.

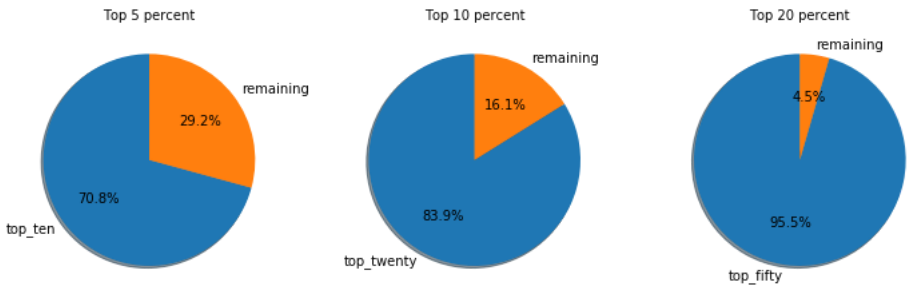
```
# get the amount sented by each user_id
user = review.groupby("user_id", as_index=False).count()
user = user.sort_values(by = "stars", ascending=False).reset_index()[["user_id", "stars"]].rename(columns={"stars":
"amount"})
```

```
def get_top_user(top):
    top_percent = user.loc[:round(user_number * top), :]
    amount = top_percent.amount.sum()
    remaining = user_number - amount
    return  (amount, remaining)
```

```
# get the lenghth 0f temp
user_number = user.amount.sum()
# get amount for top 5% users
amount, remaining = get_top_user(0.05)
top_five = {"top_ten": amount, "remaining": remaining}
# get top 10% users
amount, remaining = get_top_user(0.1)
top_ten = {"top_twenty": amount, "remaining": remaining}
# get top 20% users
amount, remaining = get_top_user(0.2)
top_twenty = {"top_fifty": amount, "remaining": remaining}
```

The Pie charts shows the comparision of number of review provided between top 5 percent user, 10 percent user, 20 percent user and remaining users. The top 5% of users provides 70% reviews of whole business, while top 10% users provides 10% reviews. If we look at the last diagram, 80% users contributes only 95% reviews. Therefore, the graph suggestes that most reviews are contributed by the minority group of users. Therefore, for local business, to get a better reviews, business could provide promotion to users who tend to write reviews. However, it is strange for user to provide more than 1000 reviews, it is possible for business to pay someone write good reviews.

```
fig, ax= plt.subplots(1, 3, figsize=(10, 10))
ax[0].pie(top_five.values(),  labels=top_five.keys(), autopct='%1.1f%%', shadow=True, startangle=90)
ax[1].pie(top_ten.values(),  labels=top_ten.keys(), autopct='%1.1f%%', shadow=True, startangle=90)
ax[2].pie(top_twenty.values(),  labels=top_twenty.keys(), autopct='%1.1f%%', shadow=True, startangle=90)
ax[0].set_title("Top 5 percent", fontsize=10)
ax[1].set_title("Top 10 percent", fontsize=10)
ax[2].set_title("Top 20 percent", fontsize=10)
fig.tight_layout()
plt.show()
```




```
# get all potential fack_user
potential_fake_user = review[review.user_id.isin(potential_fake_user_id)]
```

Then, I load user dataset to try to find potential fake users. I select the potential fake users follows the following procetures. Firstly, I select all person had zero Yelp friends, it is more possible to find fake reviewers among those groups of people. Among those groups of users, I pick up top 50 users based on the number of review they sent.

Then, I look at the fan of each users, and filter all users with the number of fans less than 5. It is abnormal for a user have sent more than 1000 reviews with no friends and fans.

Next, I define useness by calculating the ratio of useful review for all reviews, and display the top 10 potential users.

```
temp = []
with open("yelp_dataset/user.json") as file:
    for line in file:
        temp.append(json.loads(line))
user_review = pd.DataFrame(temp)
```

```
# get all user with no friends
potential_fake_user = user_review[user_review.friends == "None"].sort_values(by="review_count", ascending=False)
```

```
# get top 50 users with no friend but have most review number
top_fake_user = potential_fake_user.head(50)[["average_stars", "user_id", "review_count", "fans", "useful", "name"]]
top_fake_user.reset_index(inplace=True)
```

```
# get users with fan less than 5
top_fake_user = top_fake_user[top_fake_user.fans < 5]
```

```
# return useness of each users review
def count_usefulness(row):
    return row["useful"]/row["review_count"]
top_fake_user["usefulness"] = top_fake_user.apply(lambda row: count_usefulness(row), axis=1)
```

You might notice the name of those users is strange, indicating they pick up name randomly.

```
# top 5 poetentail users
top_fake_user = top_fake_user.sort_values(by="usefulness", ascending=True).head(10)
top_fake_user
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

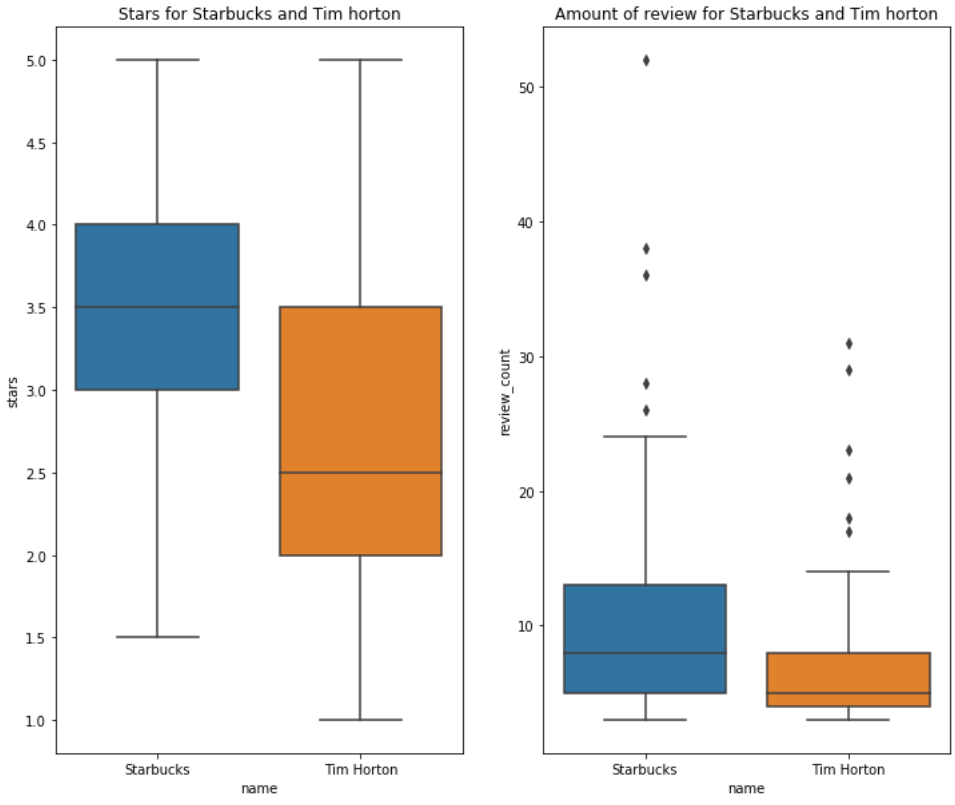
	index	average_stars	user_id	review_count	fans	useful	name	usefulness
25	315940	3.88	_nTKLtY-W7Adf9gasN4thQ	513	0	8	deCabbit	0.015595

	index	average_stars	user_id	review_count	fans	useful	name	usefulness
24	295879	3.65	iAfBmkVdzVJFz1lprDG2BQ	515	0	9	freeman_92	0.017476
30	818411	3.75	u8WpcjBhrPea9UtT7nXHmA	461	2	9	John	0.019523
6	146016	3.96	VnD9lZjizpOcioIJlTqQpQ	770	1	20	smjo	0.025974
15	1080692	3.86	qRU5enRejimk46mV2iZBlg	596	0	18	matnichol	0.030201
48	488155	2.69	lRv2dd5QLmQVSsEZLwzL2w	405	0	16	Critical	0.039506
18	935280	3.67	g5J0eySHUMOOTe0KJHLOqw	569	0	23	mabe01	0.040422
8	1244747	3.82	mtUzCYKhlc5tgyy_fZPC2A	756	0	42	Timinator	0.055556
33	604777	2.85	Qtk6D-uYIDCK9fdpAvOSPg	449	0	27	Rashaka	0.060134
14	1434945	3.72	EmoJfls864LJFa6fOjhxBQ	609	2	49	ceruleeann	0.080460

How was people's review towards Starbucks and Tim Horton's ? Previously, we exam the distribution of starbucks and Tim Horton's, but what is people's perspective towards those two companys ? Which company have higher review. Let's exam that.

Surprisingly, from the boxplot, we could observes people gave a more postive perspective towards starbucks, while a neutral perspective towards Tim Horton's. Average stars and amount of review receivced for Tim horton's is significant higher than the stars getted for Tim horton's. The average stars for Starbucks are around 3.5, while the stars for Tim horton's is around 2.5.

```
fig, ax1= plt.subplots(1, 2, figsize=(12, 10))
sns.boxplot(y="stars", x='name', data=coffee, ax=ax1[0])
ax1[0].set_title('Stars for Starbucks and Tim horton')
ax1[1].set_title('Amount of review for Starbucks and Tim horton')
ax1[1].set_ylabel("amount of review")
sns.boxplot(y="review_count", x='name', data=coffee, ax=ax1[1])
plt.show()
```



But how people define those two company, since Starbucks and Tim Horton's is not a cafe, they also offers delicious food and various beverage.

I distribute the top attributes for those two companys, following thing was interesting to notice. Firstly, almost same amount of people treat starbucks as a restaurants and a cafe. More people treat Tim horton's as a restaurants.

And not surpringly, both of them are famous by their coffee and tea and bakery. Meanwhile people in GTA area define Tim Horton's as a local Canadian company while starbucks as a American company.

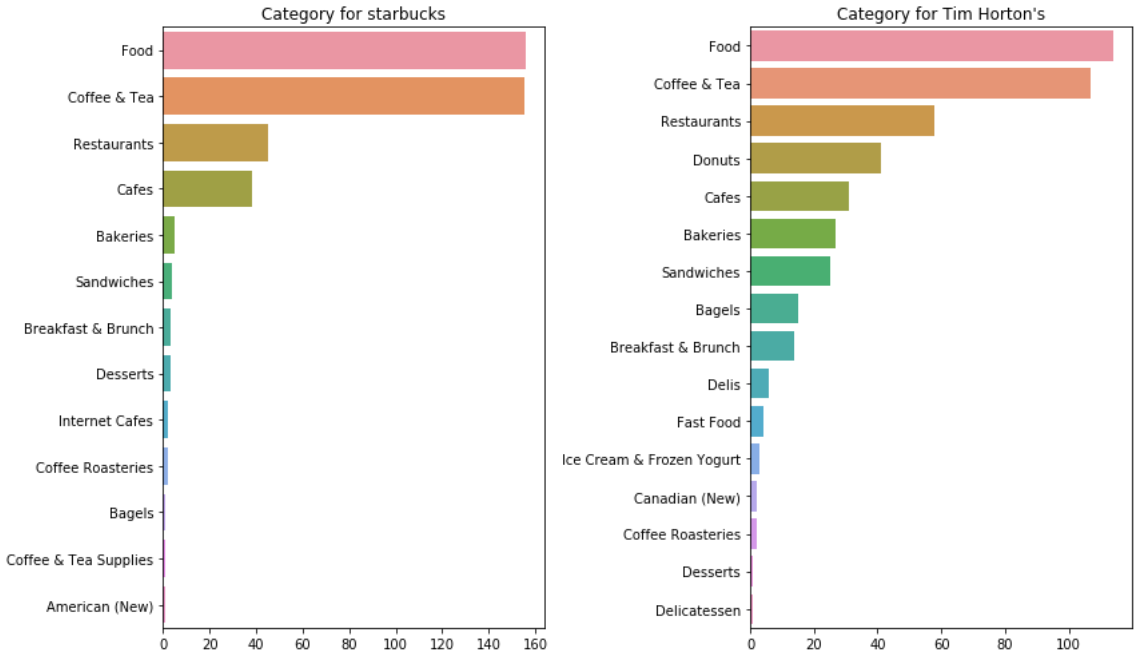
```
# get all category for starbucks
category_starbuck = starbucks.categories
starbuck = {}
```

```
for categories in category_starbuck:
    temp = categories.split(',')
    for category in temp:
        starbuck[category.strip()] = starbuck.get(category.strip(), 0) + 1
```

```
# get all categories for Tim horton's
tim_category = tim.categories
tim_to_number = {}
for categories in tim_category:
    temp = categories.split(',')
    for category in temp:
        tim_to_number[category.strip()] = tim_to_number.get(category.strip(), 0) + 1
```

```
tim_category = pd.Series(tim_to_number).sort_values(ascending=False)
starbuck_category = pd.Series(starbuck).sort_values(ascending=False)
```

```
fig, ax1= plt.subplots(1, 2, figsize=(12, 7))
sns.barplot(starbuck_category.values, starbuck_category.index, ax=ax1[0])
ax1[0].set_title("Category for starbucks")
ax1[1].set_title("Category for Tim Horton's")
sns.barplot(tim_category.values, tim_category.index, ax=ax1[1])
fig.tight_layout()
plt.show()
```



Then, I come to deal with all review texts. Firstly I extract all Starbucks and Tim Horton's business id and use the id to select reviews of starbucks and Tim Horton's. Then I create a new data frame by merging those two data frame containing information about coffee and review data frame together.

```
# extract all Starbucks and Tim Horton's business id and use for selecting reviews of starbucks and Tim Horton's
coffee_id = coffee.business_id.unique()
review_coffee = review[review["business_id"].isin(coffee_id)]
```

```
# merge coffee dataframe with review_coffee
review_coffee = review_coffee.merge(coffee, how='inner', left_on='business_id', right_on='business_id')
```

```
coffee_text = review_coffee[["name", "text"]]
```

```
def remove_punc_stopword(text):
    """
```

```
    Takes in a string of text, then performs the following:
    1. Remove all punctuation
    2. Remove all stopwords
    3. Returns a list of the cleaned text
    """
    remove_punc = [word for word in text if word not in string.punctuation]
    remove_punc = ''.join(remove_punc)
    return [word.lower() for word in remove_punc.split() if word.lower() not in stopwords.words('english')]
```

String package could be userd to remove all punctuation, For each text in reviews, I first remove all punctuation and stopwords and create a list to store each single word for future analysis purpose.

```
all_text = coffee_text.copy()
# get all text and remove the stopword from the text
all_text['text'] = all_text['text'].apply(remove_punc_stopword)
```

```
all_text.head()
```

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

	name	text
0	Starbucks	[good, management, friendly, staff, make, loca...
1	Starbucks	[variety, medium, fast, service, premises, alw...
2	Starbucks	[props, management, starbucks, frequented, dif...
3	Starbucks	[starbucks, great, come, time, study, chill, s...
4	Starbucks	[really, nice, people, run, place, good, envir...

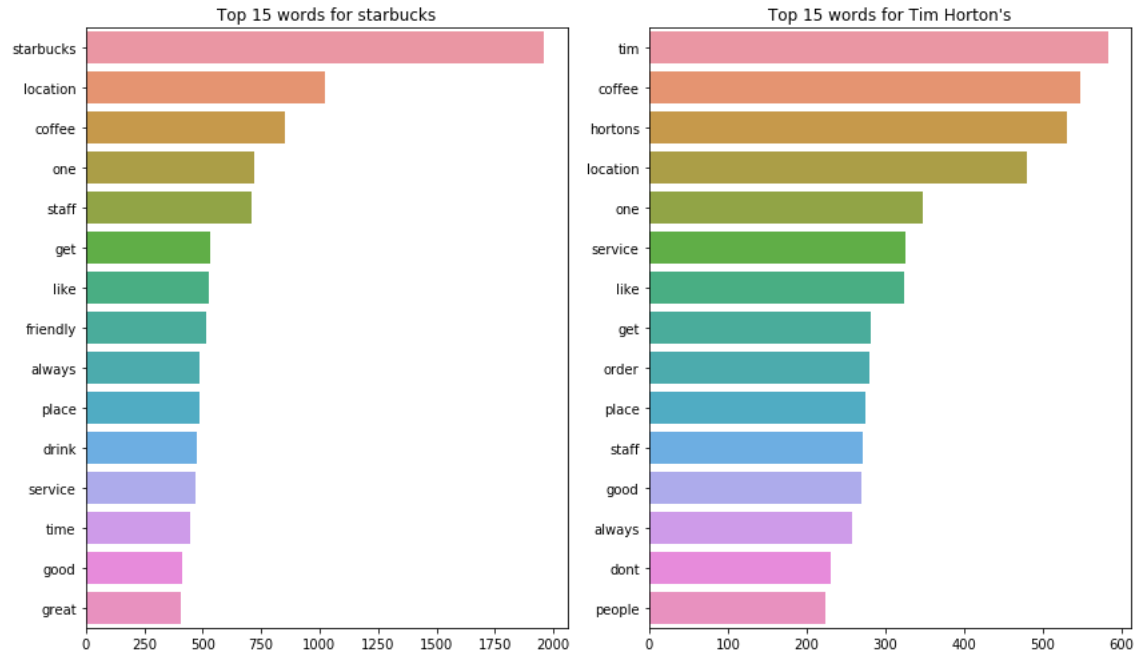
Then for each text, I iterates the who dataframe and calculate the occurence of each single word and I use bar plot to show the distribution of words.

You could notice for both starbucks and Tim horton's, the popular words are "location", "coffee", "service", "staff", indicating customer are paying attention to the location, quanlity of food and services.

```
# get top 15 words based on given business(Tim horton's or starbucks)
def get_top_word(df, business, top):
    word_number = {}
    # get all text of given business
    texts = df[df['name'] == business].text
    for words in texts:
        for word in words:
            # count the frequency of word
            word_number[word] = word_number.get(word, 0) + 1
    # get top 15 words
    top_15_word = pd.Series(word_number).sort_values(ascending=False)[:top]
    return top_15_word
```

```
top_15_word_starbuck = get_top_word(all_text, "Starbucks", 15)
top_15_word_tim = get_top_word(all_text, "Tim Horton", 15)
```

```
fig, ax1= plt.subplots(1, 2, figsize=(12, 7))
sns.barplot(top_15_word_starbuck.values, top_15_word_starbuck.index, ax=ax1[0])
ax1[0].set_title("Top 15 words for starbucks")
ax1[1].set_title("Top 15 words for Tim Horton's")
sns.barplot(top_15_word_tim.values, top_15_word_tim.index, ax=ax1[1])
fig.tight_layout()
plt.show()
```



However, which factors contribute people's decision to give a better review ? This question is interesting to consider. Therefore, it is interesting to see the vocabulary among 5 stars review and 1 stars review, by doing this, we might figure out the reason why people like or dislike.

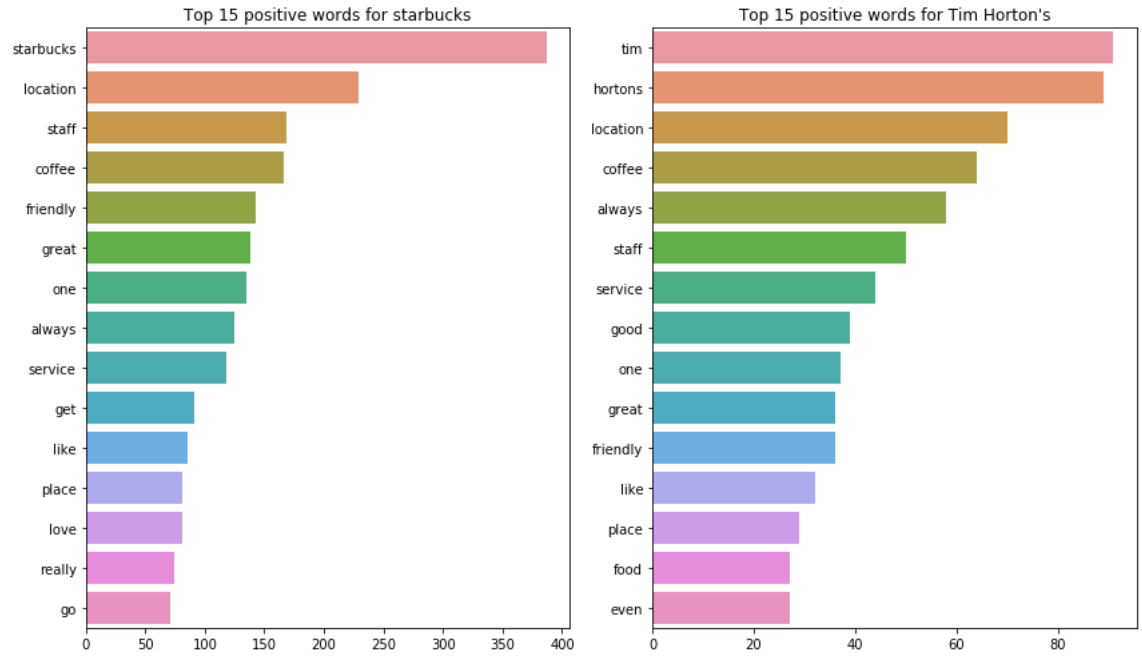
```
# get all postive review for review score 5.0 and negative review for review score 1.0
positive_review = review_coffee[review_coffee.stars_x == 5.0]
positive_review['text'] = positive_review['text'].apply(remove_punc_stopword)
negative_review = review_coffee[review_coffee.stars_x == 1.0]
negative_review['text'] = negative_review['text'].apply(remove_punc_stopword)
```

```
positive_starbuck = get_top_word(positive_review, "Starbucks", 15)
positive_tim = get_top_word(positive_review, "Tim Horton",15)
negative_starbuck = get_top_word(negative_review, "Starbucks", 15)
negative_tim = get_top_word(negative_review, "Tim Horton",15)
```

Among all 5 stars reviews, people tend to use similar language describe Starbucks and Tim Horton's. Both customers for starbucks and Tim Horton's give 5 stars due to several similar reasons:

- Location, you could notices location is the most popular words among all complishiment.
- Food: many of popular words such as coffee, food appears in the graph
- services: both staff and friendly indicates people gives 5 stars due to good services

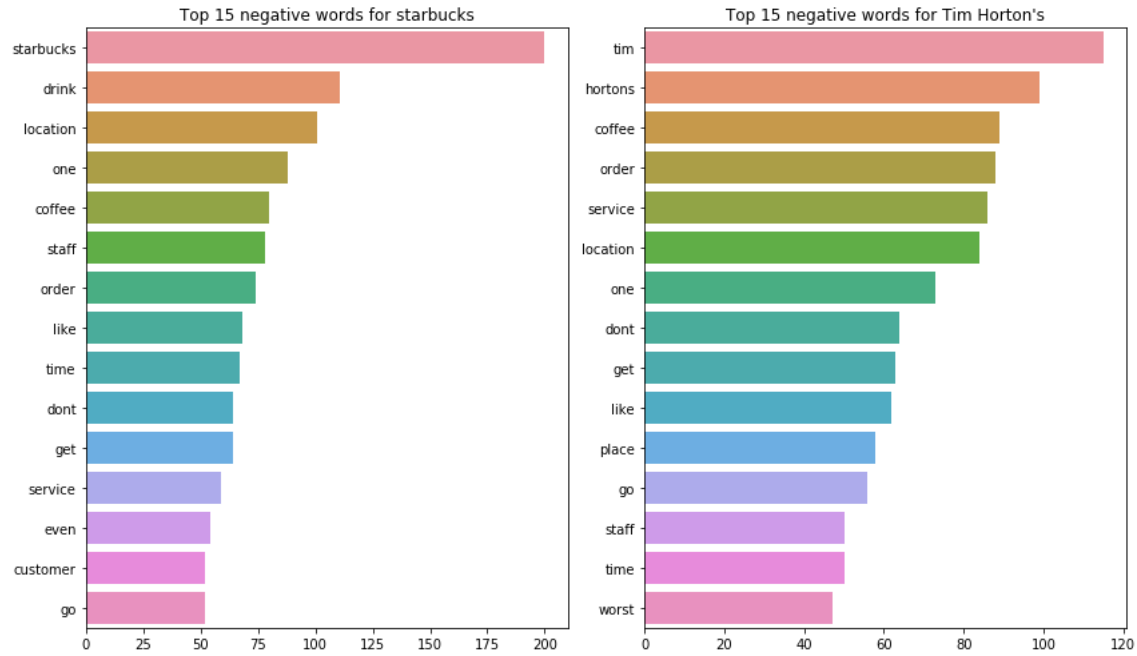
```
fig, ax1= plt.subplots(1, 2, figsize=(12, 7))
sns.barplot(positive_starbuck.values, positive_starbuck.index, ax=ax1[0])
ax1[0].set_title("Top 15 positive words for starbucks")
ax1[1].set_title("Top 15 positive words for Tim Horton's")
sns.barplot(positive_tim.values, positive_tim.index, ax=ax1[1])
fig.tight_layout()
plt.show()
```



The following graph shows the distribution of top 15 categories among 1 stars reviews, you might notice the words appearing in 1 stars review are similar to words appearing on 5 stars reviews.

In addition, people in Starbucks and Tim Horton's complains about waiting time, to improve business's rating, it is useful for starbucks and Tim Horton's to improve efficiency.

```
fig, ax1= plt.subplots(1, 2, figsize=(12, 7))
sns.barplot(negative_starbuck.values, negative_starbuck.index, ax=ax1[0])
ax1[0].set_title("Top 15 negative words for starbucks")
ax1[1].set_title("Top 15 negative words for Tim Horton's")I
sns.barplot(negative_tim.values, negative_tim.index, ax=ax1[1])
fig.tight_layout()
plt.show()
```



Therefore, For both restaurants, food and service mean everything. We can find that no matter Tim Horton or starbucks, more frequent words are food, service and some other words that are used to describe the quality of them.

Conclusion

Firstly, noticing this report mainly focus on restaurants located in GTA area, therefore, the finding of this analysis might only apply for restaurants inside GTA.

By analysing all the categories of all restaurants in Yelp reviews, indicating nightlife, bar and fast food are most popular restaurants mainly in GTA and North America.

For local restaurants in GTA which want to get better reviews. It is unecessary for local business to ask each customers to provide reviews, since the finding suggests there is no relationship between number of review recieved and the stars for each business.

However, location does have impact on restaurants rating and popularity. Previous map showing the distribution od popular restaurants and unpopular restaurants, which suggests restaurants outside downtown area have less chances to be popular. Meanwhile, analysis of reviews text among Starbucks and Tim Horton's also provide evidence to support my finding, since location is a import reason on people's perference. Therefore,for people want to open a new restaurants, it is important to consider the location of your business.

One important implication provided by this analysis is that most reviews are contributed by the minority group of users. For current restaurants, it meaningless to spend money to attract more customer to write reviews, instead, business could provide promotion to users who tend to write review, since the previous pie chart shows the minority group of people contributes to most reviews.

Most importantly, local restaurants should foucs on their food and services,

we found that for most restaurant, delicious food, and services have the key effect on people's perferences, friendly, and time are frequently appearing in people's reviews, indicating that tastes, quality of service and the efficiency might weight more than other factors and price when people are judging a restaurant. Therefore, the best way for local business to get a high review is spending money to imporve their services and food quality to which could bring customer a better dining Experiences.