Final Project

Exploring Business Units on Map

CMPT353 Summer 2020

Zirui Huang

Zhicheng Xu

Rong Li

Objective

Through exploring the OSM dataset, we found plenty of information that is related to various types of business units. Three types of business units were investigated in detail: Airbnb, Tim Hortons, and Chinese restaurants. To be specific, three major problems were proposed to solve:

1. Can the number of amenities nearby affect the rate of Airbnbs, or are there any other factors that lead to a higher or lower rate of Airbnb?
2. If I input a Tim Hortons, could I get a list of Tim Hortons similar to the one I input?
3. What could be the main factors that are affecting Chinese restaurants reviews from customers, and make predictions on reviews given some conditions.

Problem One

Problem Description:

Home-sharing is becoming popular as travelers opt for a chance to live like a local. In this problem, our ultimate target is making Airbnb recommendations for a user.
Though the recommendation itself works in a simple mechanism, a lot of work needs to be done to get the data needed to rate an Airbnb.
This recommendation would be base upon the user's current location in the city, the rate of Airbnbs in nearby neighborhoods, and the level of convenience in these neighborhoods.
Also, we are interested in exploring the relationship between the rate of Airbnbs and other factors including price and number of amenities nearby.

Data Collection & Cleaning:

The datasets we will be using for this problem are listed below, the processing and explanation of each dataset are mentioned.

(1) Open-Street Map dataset for Great Vancouver

(a) data filtering and gathering

(i) As our topic is related to traveling and accommodation, only locations with amenities related to these two areas are kept. Also, we further

categorize amenities into four categories, including sustenance (food), transportation, leisure, and arts.

(ii) The only geoinformation in the original dataset is coordinates that alone is not sufficient to solve our questions. Therefore, we used **Google Geocoding API** to gather the detailed address information of each location, including street, neighborhood, and city.

(iii) To accommodate the Airbnb dataset we collected, which unfortunately only has information for city Vancouver, we only leave locations in OSM dataset located in city Vancouver.

(2) Airbnb listings and reviews datasets for Vancouver

(a) data source

we gathered these two datasets from http://insideairbnb.com/get-the-data.html

(b) data description

(i) listings dataset: Detailed Listings data for Vancouver.

(ii) reviews dataset: Detailed Review data for listings in Vancouver.

(c) data filtering

(i) listings data

1) we removed Airbnbs which has number of reviews less than 5 or has price greater than 500.

2) columns unrelated to our problems are removed.

(ii) reviews data

1) Airbnbs that are filtered out in the above part are removed.

(3) Tweets dataset

(a) data source

we gathered this dataset from

https://www.kaggle.com/rtatman/analyzing-multilingual-data/execution

(b) dataset description

This dataset contains 10,502 tweets, randomly sampled from all publicly available geotagged Twitter messages, annotated for being in English, non-English, or ambiguous in language.

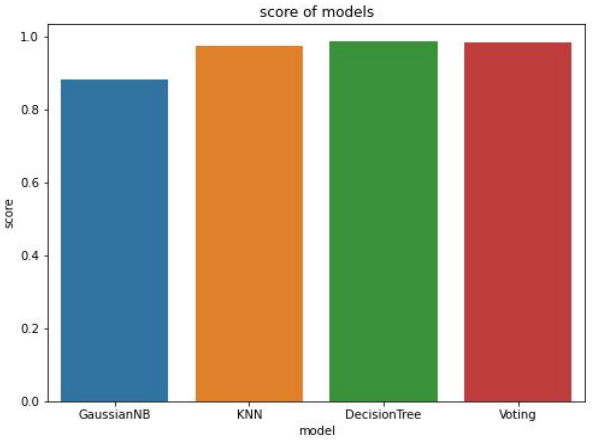
(c) data filtering:

(1) Tweets that are ambiguous in language are removed. So remaining tweets are either definitely in English or not in English.

(2) Some preprocessing on those textual data are made, including making all letters lowercase, removing punctuation and stop words.

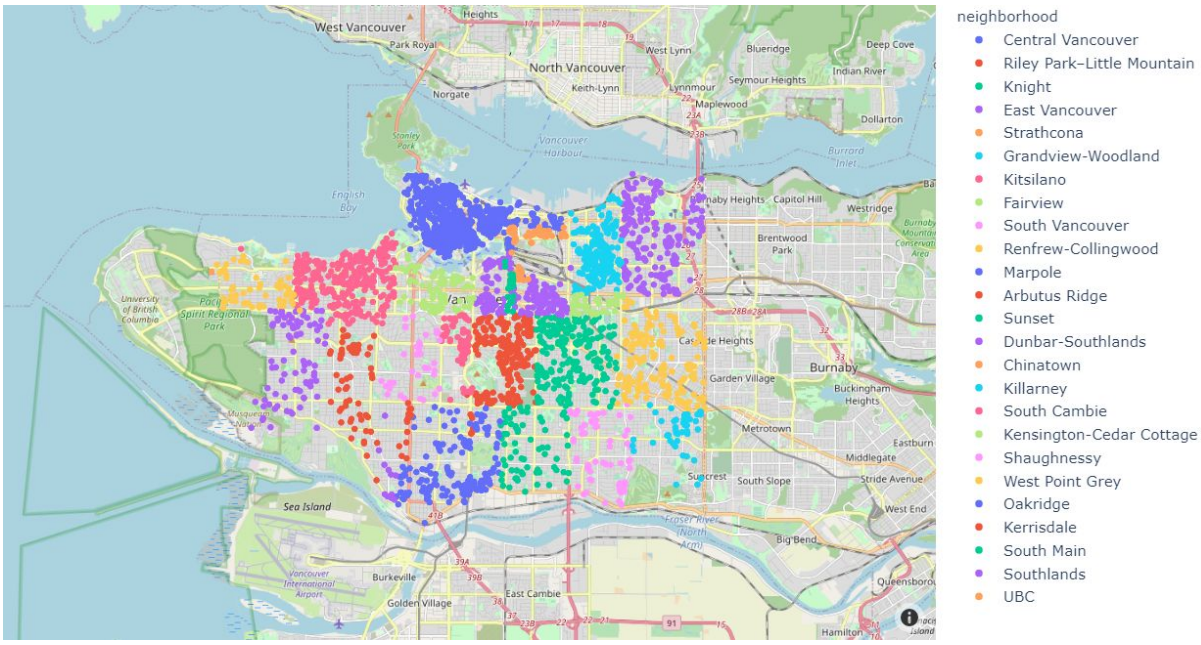
Predict Neighborhood for Airbnbs

Because our analysis would be based on neighborhoods, we want to know which neighborhood each individual Airbnb located. Although the original airbnb dataset has neighborhood entry, unfortunately it does not perfectly match the one in OSM data. What we are going to do is fitting a model based on coordinates to predict the neighborhood of Airbnbs. The training data is the excerpt from the Vancouver OSM data with labeled neighborhood.



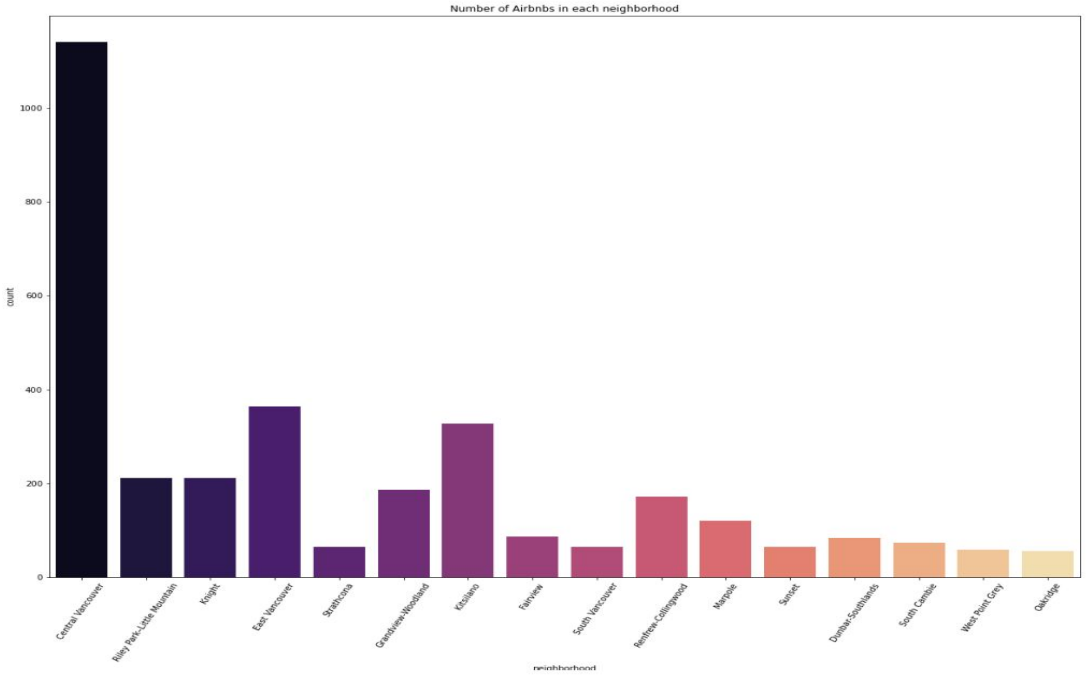
The models we tested are GaussianNB, KNeighborsClassifier, DecisionTree-Classifier, VotingClassifier. It seems all of these models except GaussianNB work pretty well.

Below is a visualization of the result of decision tree model, we can see Airbnbs are neatly classified into different neighborhoods. (some color ambiguities here due to a large number of classes)



From figure above, we can see there are some neighborhoods in the Airbnb listings dataset that only has few Airbnbs. Therefore, we remove those Airbnbs from neighborhoods that have less than 50 Airbnbs.

Now, we can see the distribution of Airbnbs in the city. Central Vancouver, East Vancouver, Kitsilano, Riley Park, and Knight have a high density of Airbnbs.

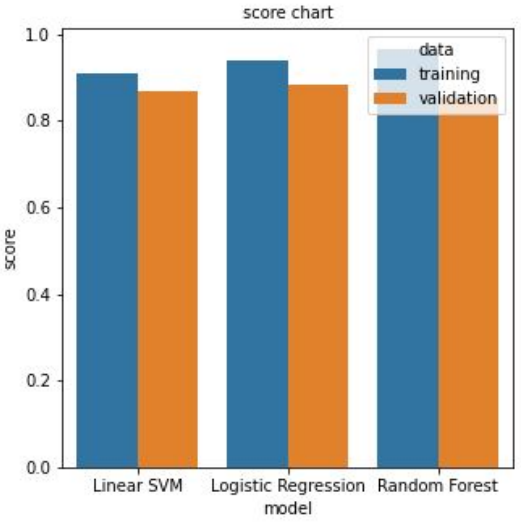


Rating Airbnb

The original Airbnb review dataset does not provide rates in numerical value, we are trying to give each individual Airbnb a rate based on its reviews.

First, not all reviews are in English, so the first step we are going to do is predict the language of each review and only keep those that are predicted to be written in English.

1. Feature Extraction (Count Vector)
    a. We transform tweets from our training dataset into an array having the count of appearance of each word in it. The intuition here is that the text that is written in the same language (English or not English) may have the same words repeated over and over again.
2. Train Our Models
    a. With the numerical representations of the tweets ready, we can directly use them training data for some classic machine learning models
    b. Here, the three classical machine learning models we tested are Linear SVM, Logistic Regression and Random Forest Classifier.
    c. The scores of those three models are visualized below.
    d. The overall performance is good with count vectors. The best model, **logistic regression** achieved up to 0.8841% accuracy with validation data. The reason for this good performance might be because of the nature of this specific dataset where the language of the text is heavily dependent on the presence of some specific words [1].



3. Make prediction
    we then predict the language of reviews in our Airbnb dataset, some sample results are shown below.

	comments	english
0	Hemos estado 7 días en la casa de Rami. A pesa...	0
1	Wir waren nach einer 4wöchigen Rundfahrt die l...	0
2	Großartige Unterkunft, wunderbar gelegen bezüg...	0
3	Vielen Dank für den tollen Aufenthalt in Vanco...	0
4	Unterkunft sehr gut ausgestattet und sauber, s...	0

	comments	english
0	this accommodation was excellent. beautiful sp...	1
1	The host canceled my reservation 13 days befor...	1
2	This apartment is fantastic, just what I and m...	1
3	Very nice apartment and great view. Close to S...	1
4	Both Rami and Mauricio made our family of 5 fe...	1

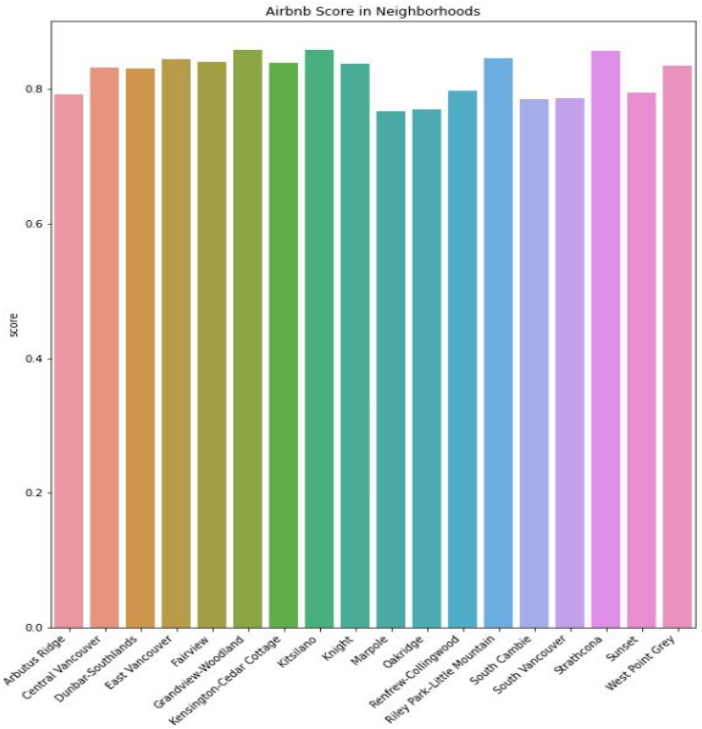
4. Sentimental Analysis of Review

We used sentiment intensity analyzer **VADER** to evaluate each review.

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. With VADER, we can get the Positivity and Negativity score of each review sentence as well as how positive or negative it is [2].

With the compound score from VADER to give a numerical rating to each individual review. The compound score is a metric that calculates the sum of all the lexicon ratings which have been normalized between -1 (most extreme negative) and +1 (most extreme positive)

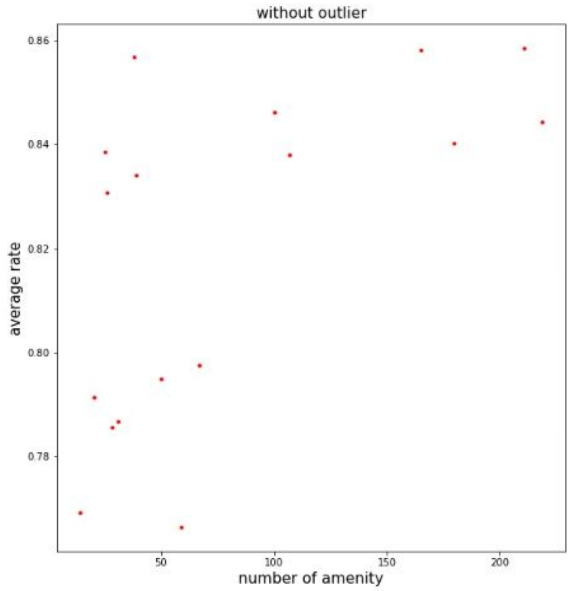
With the approximated average rate of each Airbnb, we can have a general sense of how the average rate of Airbnbs distributed in each neighborhood.

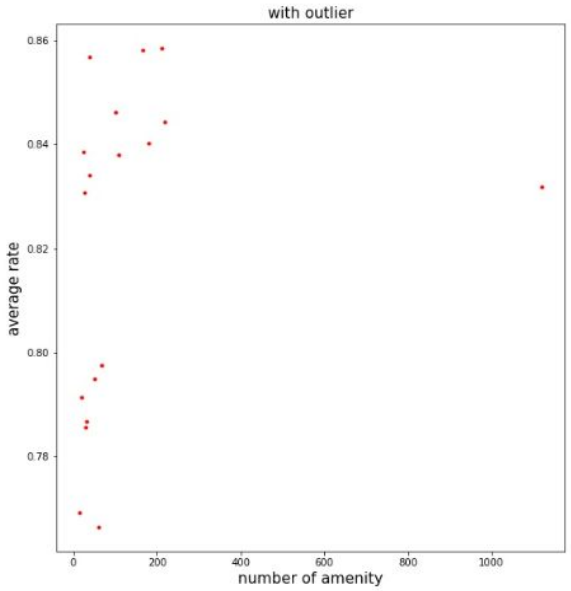


Statistic Analysis

With all the data we have obtained, we can now explore the relationship between Airbnb score and other factors. Most of our data are not perfectly normally distributed, thus our analysis would be based on central limit theorem such that we launch our analysis with sample means of each neighborhood.

1. relationship between the rate and number of amenities nearby.

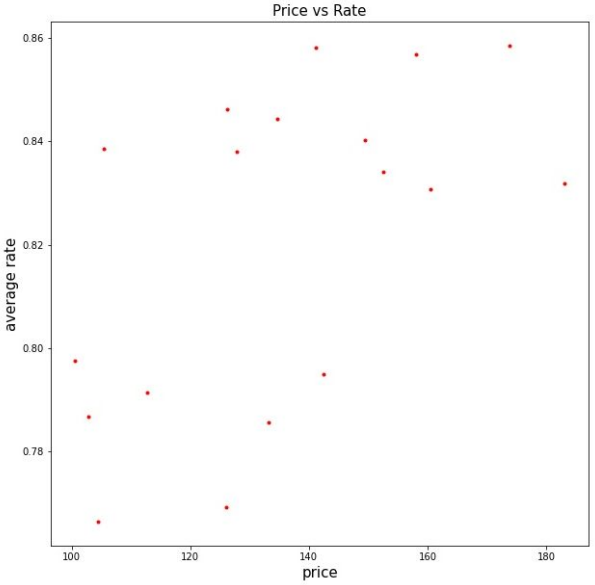




We expect there is a linear relationship between rate and number of amenities.
We can find a linear relationship from the plot if we remove the outlier (Central Vancouver). Also, the p-value of the test for the regression slope is 0.01141, which is smaller than the usual α value. Therefore, we can conclude that rate does depend linearly on number of amenities nearby.

2. relationship between the rate and price

The p-value of the regression slope test is equal to 0.01367, so we can reject the null hypothesis and conclude that there is a linear relationship between price and rate such that higher price can result in a higher rate.



Airbnb Recommendation

Finally, with all the data we have obtained, we can recommend Airbnb to users.
The input of our program is a photo with EXIF data that contain geographic coordinates.

The program can help user to find three most well-reviewed Airbnb in each nearby neighborhoods. The details of recommended Airbnbs are provided.

Reference:

1. *Xq. (2020, May 23). Applied Machine Learning: Part 3. Retrieved from https://medium.com/the-research-nest/applied-machine-learning-part-3-3fd405842a18*
2. *Pandey, P. (2019, November 08). Simplifying Sentiment Analysis using VADER in Python (on Social Media Text). Retrieved from https://medium.com/analytics-vidhya/simplifying-social-media-sentiment-analysis-using-vader-in-python-f9e6ec6fc52f*

Limitations

1. The Airbnb data we obtained is limited in geography range. This makes our recommendation can only work in city of Vancouver. If more time is allowed, or more advanced web crawler technique can be mastered, we may try to obtain Airbnb information just from the official Airbnb website.
2. The mechanism recommendation is simple. If a wider range of category of locations, can be obtained from OSM data, such as shopping center and tourism point, we may make better Airbnb recommendations according to user's preference.
3. There can be ways to further enhance the accuracy score of our language prediction model. Some better preprocessing techniques and more advanced machine learning models like deep learning help to generate better result.
4. There can be lurking variable affect the relationship between airbnb rate and the factors we examined.

Problem Two

Problem Description:

I a fan of Tim Hortons since it is affordable, and it has many branches around the city. I like to go to the stores with large parking spaces and surrounded by different kinds of restaurants. For example, the one at SFU and Metrotown is a good choice for me to grab a coffee. I do not like the Tim Hortons stores inside the gas station or located in a sparsely populated area since they have less variety of products and sometimes the donuts I liked sold out very quickly. I will get a list of Tim Hortons stores near me when I opened the Google Map but I have to click the stores to see if that one matches my preference. Therefore, I am wondering if it is possible to find some similar Tim Hortons stores based on the one I usually go to, and then I have more options across the city!

Data Collection & Cleaning:

I used the OSM dataset given in the project and the Tim Hortons dataset downloaded from the Kaggle which contains the basic characteristic of the stores such as if the store has its Wi-Fi. Firstly, I need to combine OSM and TH dataset. I first get all the Tim Hortons **(TH)** stores in the OSM dataset. Then I filter out all the TH that are not located within Vancouver in the TH dataset which reduces to 195 stores. Then I calculated the distance between every store in the OSM dataset and TH dataset based on the latitude and longitude. I connected the store in the OSM dataset to the closest TH store in the TH dataset based on the distance I just calculated. After that, I dropped out the stores in OSM if its closest distance to the stores in the TH dataset is larger than 200 meters since it may be because the data was missing or there are some duplicates and I did find two duplicate records inside the TH dataset. After this join operation, I got a new data frame that merged with OSM and TH dataset and there are 112 stores in total.

Secondly, I need to convert all the categorical variables in the TH dataset into numerical values since it is hard to find the similarity in String value and String will make the distance between points even harder to calculate. I used preprocessing.LabelEncoder which is in the sklearn package to encode the category between 0 and number of categories – 1. For example, I used this to encode the column 'driveThruLaneType' and it converted 'DT' to 0, 'DT Only' to 1, 'No DT' to 2. I also handled some obvious incorrect data in the TH dataset. For example, there are several 'DDT' records in column 'driveThruLaneType' and I just changed them into 'DT'. Still, there are some missing records for the parking types column. I just filled all of the missing records into a new category and encode them as 0. This is reasonable since their parking type is missing is probably because the stores are not popular and thus it is harder to get that information, and this could be considered as a similarity. After that I dropped some unnecessary columns such as 'hasDriveThru' is highly correlated with 'driveThruLaneType', all the stores' 'hasCatering' column has the same value, etc. I also extracted the restaurant ID from the data since I suspect the ID has some correlation with the foundation date of that TH like older TH may have a smaller restaurant ID. Figure 1 is a glimpse of the data frame after I did the data cleaning.

	amenity	_id	driveThruLaneType	frontCounterClosed	hasBreakfast	hasCurbside	hasDineIn	hasDelivery	hasMobileOrdering	hasTakeOut	hasWifi	mobile
0	0	67542	2	0	0	0	0	1	0	1	0	
1	0	68278	0	0	1	0	0	1	1	1	1	
2	0	67920	2	0	1	0	0	1	1	1	1	
3	0	68011	0	0	1	0	0	1	1	1	1	
4	0	75916	2	0	0	0	0	1	1	1	0	

Figure 1 - dataset merged with Tim Hortons and OSM

Feature Engineering & Feature Scaling:

The dataset I got from Figure 1 can only reveal the basic features of the TH, however, I like the TH in the Metrotown is not only because it has a large parking lot and dining space but also it is located at the Metrotown and there is a lot of great food nearby. Someone like that TH maybe it is because there is a post office nearby and he can grab a coffee after he picks up the package. Therefore, I need to add more features that are related to the environment to get a better prediction. Firstly, I calculated the length of 'tags' in the OSM dataset and the intuition behind this is that a bigger TH should have more tags associated with it. Therefore, tags length could somehow signify the scale of the TH. After I added the features which may

help me to find the TH with a similar scale, I need to find the amenities closest to that TH. Then I unearth all the amenities in OSM that are within 20 meters around the TH and select the top three amenities (the popularity is based on the length of those amenity tags). For example, one of the reasons I prefer to go to TH at Metrotown is because there is a McDonald's near it and that McDonald's should be in the top three amenities list since McDonald's should have a fair number of tags related to it in the OSM dataset. For the OSM dataset, I dropped out all the rows if their name is null because most of these rows are boring. For example, I found out some less popular TH which does not have too many amenities near it will choose ‘waste_basket’ as its top three amenities and this is not significant for our prediction since the wastebasket is not likely the things I will notice when I go to a TH and every TH should have a wastebasket. Figure 2 shows the general structure of the top three amenities columns and these amenities should likely be the things I am interested in when I go to that TH.

amenity0	amenity1	amenity2
pharmacy	fast_food	bank
restaurant	fast_food	restaurant
restaurant	fast_food	veterinary

figure 2, amenity columns

After I did the data featuring which I added the length of tags and the top three amenities near the TH, I got 18 columns of data and some of them have a different scale like restaurant ID. Then I made a pipeline for my model which will first use StandardScaler to normalize the data and then applied Principal component analysis (PCA) to reduce the dimension. I choose StandardScaler is because most of my data did not have an outlier such as ‘hasDriveThru’ is either 0 or 1, most of the restaurant id is 5 digits and start with 6, etc. Then I can use StandardScaler to compute the empirical mean and standard deviation to scale the data to unit variance. I tried to use MinMaxScaler before, but it did not produce a good result. I guess the reason is that most of my columns are 0 and 1 values, then these true values will dominate the similarity distance calculation and then other columns like restaurant ID, amenity columns have less impact since their values are between 0 and 1 after the MinMaxScaler scaler. After

that, I applied the PCA to reduce the dimension to 5 in order to avoid the curse of dimension. Because the KNN and KMEAN algorithms don't work well with high dimensional data. With a large number of dimensions, it becomes unmanageable for the algorithm to calculate the distance in each dimension.

_id	driveThruLa	frontCounte	hasBreakfas	hasCurbside	hasDineIn	hasDelivery	hasMobileO	hasTakeOut	hasWifi	mobileOrde	addr	parkingTyp	tags_len	amenity0	amenity1	amenity2
69555	2	0	1	0	0	1	1	1	1	1	4700 KINGSWAY BURNABY, British Columbia V5H 4N2 - Canada	1	15	fast_food	cafe	fast_food
69851	2	0	1	0	0	1	1	1	1	1	#1011 10355 152nd STREET GUILDFORD TOWNE CENTRE SURREY, B	1	7	fast_food	fast_food	fast_food
70110	2	0	1	0	0	1	1	1	0	1	467 INTERURBAN WAY VANCOUVER, British Columbia V5X 0C5 - C	0	8	cafe	fast_food	empty
68552	2	0	1	0	0	1	1	1	0	1	4820 KINGSWAY ST BURNABY, British Columbia V5H 4P1 - Canada	0	7	fast_food	fast_food	fast_food
68584	2	0	1	0	0	1	1	1	1	1	10320 152 STREET SURREY, British Columbia V3R 4G8 - Canada	1	17	restaurant	empty	empty

Figure 3 result for KNN with the input TH located at Metrotown

_id	driveThruLa	frontCounte	hasBreakfas	hasCurbside	hasDineIn	hasDelivery	hasMobileO	hasTakeOut	hasWifi	mobileOrde	addr	parkingTyp	tags_len	amenity0	amenity1	amenity2
68552	2	0	1	0	0	1	1	1	0	1	4820 KINGSWAY ST BURNABY, British Columbia V5H 4P1 -	0	7	fast_food	fast_food	fast_food
68556	2	0	1	0	0	1	1	1	1	1	4700 KINGSWAY BURNABY, British Columbia V5H 4N2 - Ca	1	15	fast_food	cafe	fast_food
68851	2	0	1	0	0	1	1	1	1	1	#1011 10355 152nd STREET GUILDFORD TOWNE CENTRE SU	1	7	fast_food	fast_food	fast_food

Figure 4: result for KMEAN with the input TH located at Metrotown

driveThruLa	frontCounte	hasBreakfas	hasCurbside	hasDineIn	hasDelivery	hasMobileO	hasTakeOut	hasWifi	mobileOrde	addr	parkingTyp	tags_len	amenity0	amenity1	amenity2
2	0	1	0	0	1	1	1	0	1	6525 OAK ST VANCOUVER, British Columbia V6P 3Z3 - Canada	1	1	empty	empty	empty
2	0	1	0	0	1	1	1	0	1	2177 DOLLARTON HWY - ESSO OTR NORTH VANCOUVER, British Columbia V7I	1	0	empty	empty	empty
0	0	1	0	0	1	1	1	1	1	2711 192ND ST SURREY, British Columbia V3S 3X1 - Canada	1	4	fast_food	empty	empty
0	0	1	0	0	1	1	1	0	1	6036 GLOVER RD, UNIT 6015 LANGLEY, British Columbia V2Y 2P3 - Canada	1	9	car_wash	empty	empty
0	0	1	0	0	1	1	1	0	1	7195 KING GEORGE HWY SURREY, British Columbia V3W 5A4 - Canada	1	7	empty	empty	empty

Figure 5: result for KNN with the input TH located in the gas station

Data analysis:

After I transformed the dataset (PCA & StandardScaler), I applied the KNN (metric = ‘manhattan’) and KMEAN(n_clusters=24) to find a similar THs with a targeted TH. Figure 3 above illustrated the result I gained from the KNN model and I choose the TH at Metrotown as the targeted TH. I applied the KNN model and it basically returns the 4 neighbors that near my targeted TH. Since I did not have labels in my dataset to say that if that TH is similar to others, I did not use KNN to predict anything and I just used the model to look for the 4 closest neighbors. As you may notice from Figure 3, none of them have drive-through (2 for no drive-through) and all of them have breakfast, delivery, mobile order, and take-out services. More importantly, these TH are usually surrounded by fast food, café, and restaurant. Also, one of the TH that KNN returned is the one in the Metrotown food court (the 4th) which makes sense since it has a similar environment as the targeted TH.

As for KMEAN, the number of the TH it returns is not fixed like a cluster can have only one or many data records. But the result I got from KMEAN model are all included in the KNN model and I tried several times, the result that KMEAN returned always has some overlapped with KNN’s result. Figure 4 shows the result gained from the KMEAN model with the targeted TH at Metrotown, it is essentially the subset of the result from the KNN model. Then

I picked another TH located inside the gas station and want to see if the model can find the TH in this similar location. The result was given in Figure 5 and we can conclude that delivery and take out are the must-have services and parking type must be shared (2 for shared). From the amenity columns, we know that these TH are located in a quiet place and most of them are in the gas stations based on the 'addr' column. Therefore, I think my models did a decent prediction and I can use these models to search for some similar TH.

Limitations:

1. Before I added the top three amenities, the models did not produce a decent result. The nearby amenities did have a great effect on the prediction, but it would be even better if I could get the reviews for the TH and vote the top three amenities based on both ratings and tags length. Because the rating could also imply some similarity and the number of people rated the TH would be a better estimator for the popularity.

2. The operation time of TH in the dataset is messy and many of them are missing. This is a huge loss since operation time could be useful for our prediction. I cannot find a good way to impute the missing values since it is not plausible to assume it closed at 7 if most of TH are closed at 7. The ones operated 24 hours should be the most interesting one and we cannot impute it based on the majority. If I have more time, I think I can train a model to predict the operation time.

Problem Three

Problem Description:

As a culturally diverse country, there are lots of Asian restaurants in Canada, as we can see on the OSM map as well. We are interested in what could be the main factors that are affecting Chinese restaurants reviews from customers, and potentially make some persuasive predictions on how well a newly opened Chinese restaurant would perform given some factors.

Like all other types of restaurants, Chinese restaurants are reviewed by some of the common factors as well, such as parking spot, delivery system, noise level, price range, family-friendly, or outdoor seatings. As a normal consumer, people first look at these features out of a restaurant. Now when it comes to Chinese restaurants, especially in a western country, different people may have different preferences. Based on our observations, we suspect people from different cities may have vastly different opinions on Chinese cuisines. For example, we believe there should be a big difference in how people would review a Chinese restaurant in general between a Chinese population heavy city and a French population heavy city. Of course, this is all based on our guess, being in different Canadian cities may or may not affect the overall reviews. Therefore, we are going to use statistical models to retrieve the potential relationship between the two.

Data Collection & Cleaning:

To test out whether there is a relationship between the level of reviews and geographic location(cities in this case) of the restaurants, we would need a dataset of restaurants from different cities all over Canada, along with a dataset of customer reviews. Our OSM dataset satisfied the first, in which it has all the restaurant information along with their geographic locations, however, there are no customer reviews available for us. Therefore, we needed to find another dataset online that has customer reviews. We found the Yelp database contains every customer reviews on business units including Chinese restaurants, however, for some reason, Yelp did not have data of customer reviews in BC alone. We couldn't link this Yelp database to our OSM datasets. Fortunately, Yelp has another dataset that has very similar data as the OSM dataset, in which it has all the information about Chinese restaurants that are

being reviewed, including geographic information, tags(features of restaurants), and business id to join to the reviews dataset.

We have retrieved two datasets from Yelp, “yelp_academic_dataset_business” contains information of business units from different cities, while “yelp_academic_dataset_review” contains customer reviews of business units online at yelp.com. Sample rows of these two datasets are as follow:

business_id	name	address	city	state	postal_code	latitude	longitude	stars	review_count	is_open	attributes
P0c_u3g-3LiWHpb2XnOQ	Lee Cafe	1046 Regent Pkwy, Ste 107	Fort Mill	SC	29715	35.075476	-80.933904	4.0	267	1	{'NoiseLevel': 'u'average", 'RestaurantsReser...
pH6ZuCQY3o1yBuJ67PQ	Master Mix HotPot	261 Spadina Avenue	Toronto	ON	M5T 1H1	43.652251	-79.397642	3.0	17	0	{'RestaurantsDelivery': 'False', 'Ambience': '...

business_id	stars	name	city	categories
dvbcUnKv2awsIxog7dO4vw	4	Hong Kong Bistro Cafe	Toronto	Cafes, Chinese, Restaurants
dvbcUnKv2awsIxog7dO4vw	3	Hong Kong Bistro Cafe	Toronto	Cafes, Chinese, Restaurants

The customer review dataset is too big to run against Pandas, which contains 6Gbs of data. The dataset is simply too big to fit in the memory of our laptops. Fortunately, not all reviews in the dataset are needed, in our case, we selected 6 major cities from Canada to do comparisons, therefore, we would only need reviews that came out of those 6 cities. In addition, we didn’t need reviews of all other business units besides Chinese restaurants only. The final resulting review dataset should be trimmed small enough to comfortably fit into our local memory. We decided to read chunks of that data separately into memory and process pieces of data separately. In particular, we are setting **chunksize=50000** when reading the review dataset. Pandas will only read 50000 rows at a time into memory which won’t cause memory insufficient problems. When processing each chunk, we simply filtered out all other reviews of business units except Chinese restaurants in the selected 6 cities. For each chunk, after trimming down the size, that chunk will be stored into a data frame.

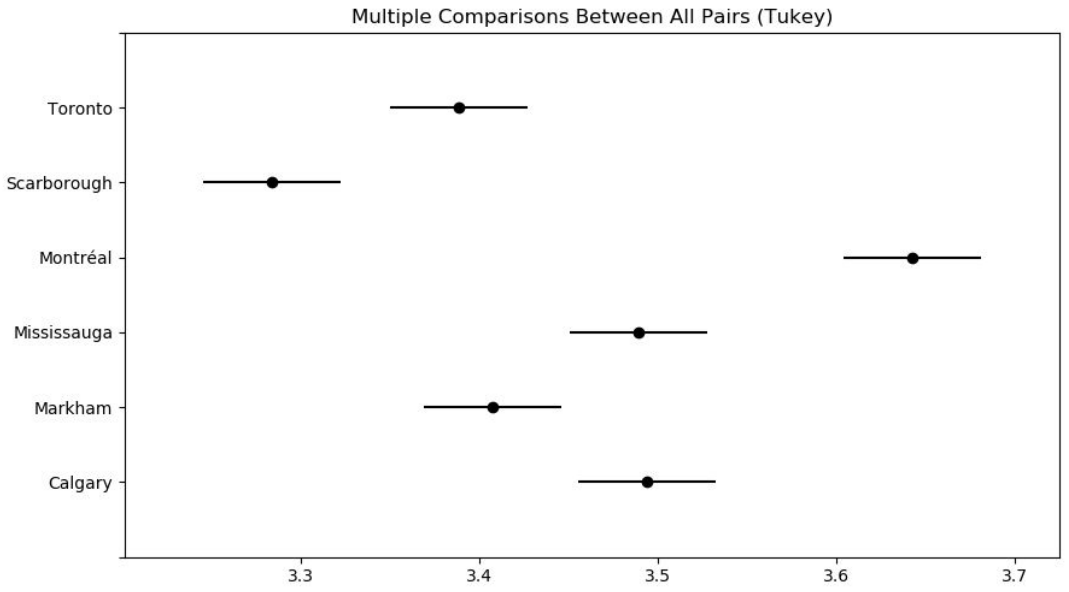
Statistic Analysis:

After trimming down the review dataset, we chose to do a pairwise comparison between the selected 6 cities by using Post Hoc Analysis to find out whether reviews in these cities have different means. The chosen 6 cities are Toronto, Calgary, Mississauga, Markham, Scarborough, and Montréal. These 6 chosen cities are physically scattered around Canada. They all have a sufficient amount of Chinese restaurants and customer reviews to the statistical test. The null hypothesis in our test is: the mean of reviews in two cities are the same. As long as we can reject this null hypothesis, we can conclude the alternative hypothesis succeeds, in which there is a mean difference between different cities. Here is the returned post hoc result:

Multiple Comparison of Means - Tukey HSD, FWER=0.05

group1	group2	meandiff	p-adj	lower	upper	reject
Calgary	Markham	-0.0866	0.0177	-0.1639	-0.0093	True
Calgary	Mississauga	-0.0044	0.9	-0.0817	0.0729	False
Calgary	Montréal	0.149	0.001	0.0717	0.2263	True
Calgary	Scarborough	-0.2103	0.001	-0.2876	-0.133	True
Calgary	Toronto	-0.1052	0.0015	-0.1825	-0.0279	True
Markham	Mississauga	0.0821	0.0296	0.0048	0.1595	True
Markham	Montréal	0.2356	0.001	0.1583	0.3129	True
Markham	Scarborough	-0.1237	0.001	-0.201	-0.0464	True
Markham	Toronto	-0.0187	0.9	-0.096	0.0587	False
Mississauga	Montréal	0.1534	0.001	0.0761	0.2307	True
Mississauga	Scarborough	-0.2058	0.001	-0.2831	-0.1285	True
Mississauga	Toronto	-0.1008	0.0028	-0.1781	-0.0235	True
Montréal	Scarborough	-0.3592	0.001	-0.4365	-0.2819	True
Montréal	Toronto	-0.2542	0.001	-0.3315	-0.1769	True
Scarborough	Toronto	0.105	0.0015	0.0277	0.1823	True

Based on the resulting table, 15 pairs of comparisons are made between the selected 6 cities. 13 out of 15 pairs rejected the nun hypothesis. The other two we could not conclude base on the result of failing to reject the null hypothesis. In the other words, almost 90% of the pairwise comparison successfully rejected the null hypothesis, which we could use as a proof of our prediction that people from different cities may have different opinions on Chinese cuisines and overall give higher or lower reviews compared to some other cities. For a visual representation, we have plotted a spiffy plot of the confidence intervals of each means:



This plot once again visually proved there is an underlying difference between the mean of reviews in different cities.

Prediction on Reviews:

Now that we have proved there is a big possibility that being in different cities, Chinese restaurants could receive different levels of satisfaction from customers. In other words, geographical difference plays an important role in customers' reviews towards Chinese restaurants. We wanted to use this relationship along with some other features of a Chinese restaurant to predict the future overall level of review it could receive from customers. In particular, besides located cities, 7 features will be taken into consideration: delivery system, noise level, price range, does the restaurant have a take-out system available, is the restaurant environment family-friendly, does it have outdoor seatings and does it have a TV. These features are available as tags in the original business units dataset. After trimming down the dataset, the resulting dataset shows:

business_id	name	city	stars	delivery	NoiseLevel	PriceRange	TakeOut	GoodForKids	utdoorSeatin	HasTV
u3g-3LiWHpb	Lee Cafe	Fort Mill	4	FALSE	u'average'	2	TRUE	TRUE	TRUE	TRUE
ZuCQY3o1yB	ter Mix Ho	Toronto	3	FALSE	'average'	2	FALSE	TRUE	FALSE	FALSE

Our original plan was to use the Linear Regression model to predict the review. The first thing we need to do is to do some data transformation to use any Machine learning models. In our dataset, city, NoiseLevel, and PriceRange are represented as Strings, while delivery, Takeout, GoodForKids, OutdoorSeating, and HasTV are all booleans. To fit them into our linear regression model, they need to be numeric. To achieve that transformation, we used LabelEncoder to classify the strings and booleans into integers. It is worth mentioning that our output of training and prediction is the 'stars' column, which is now a float type between 1.0-5.0 representing the average reviews that restaurant received from its customers. After fitting the training set into our Linear Regression model, we have received a 0.006 score from the validation set, which is unacceptable.

We then decided to turn the problem into a classification problem. Since the average reviews can only be a float with a single decimal place which makes only 5 * 10 = 50 possible outcomes when it comes to customer reviews. We decided that turning these 50 possible float outcomes into 50 categories isn't tedious so that we could use more useful models like Random Forest, GaussianNB, or K Neighbors Classifier, etc. We once again used LabelEncoder to do such transformation. After the transformation is done, we used the Random Forest model to train our dataset and predict reviews as categories. As a result, we got 0.327 as the score of validation, which is still unacceptable, however, much better than the first Linear Regression model that we have done.

Problem Revisions:

We wanted to improve on getting more accurate predictions. It seems having geographic locations and some other features are not sufficient to accomplish a good prediction on an average review that the restaurant would receive and being accurate to single decimal space numbers. We decided to modify the output that we are trying to predict i.e. the average review so it is easier for our model to predict. We decided to transfer reviews from being single decimal space numbers to integers. So that there would be only 5 possible outcomes instead of 50. Though such transformation is technically losing information, we could make better predictions out of it. The Random Forest model is used once again on the modified dataset, and a score of 0.615 was received as a result, which bumped up 30% from last time.

After we successfully increased our validation score by modifying the training set, we wanted to use the same concept to get an even better score. This time, we narrow down the range of predication further, reviews are changed from being integers of 1-5 to booleans. In particular, we changed the question a bit: given geographic location, and other features, predict how customers would review the restaurant in terms of "good" or "bad". Obviously, the problem became much simpler, since it changed from having 5 possible integer outcomes to 2 boolean outcomes. We wrote a function to determine in what range the reviews are treated as "good" and in what range the reviews can be treated as "bad". In particular, reviews less than or equal to 2.5 stars are classified as "BAD" and reviews greater than 2.5 stars are classified as "GOOD". The Random Forest model was used as well. As expected, a good validation score of 0.868 was received from the model. We can now confidently predict a review(good or bad) of any given restaurants with given conditions. The script output of our four revisions was:

```
LinearRegression model predict score(treat reviews as numerics):  0.005744653261912758
RandomForestClassifier model predict score(reviews as categories):  0.32730263157894735
RandomForestClassifier model predict score(predict integer reviews):  0.6151315789473685
RandomForestClassifier model predict score(predict good/bad reviews):  0.868421052631579
```

The result of validation scores changed dramatically as we modified our dataset. In summary, our four revisions are as follows:

Revision #	Model used	Target of predication	Target type	Possible outcomes	Average score of validation
1	Linear Regression model	float between 1 to 5(one decimal)	Numeric	50	0.005744653
2	Random Forests Classifier	transfer to categories from above	Categories	50	0.327302632
3	Random Forests Classifier	integer between 1 to 5	Categories	5	0.615131579
4	Random Forests Classifier	boolean("GOOD" or "BAD")	Categories	2	0.868421053

It turned out we couldn't accurately predict the specified review, which is the one decimal place float number between 1 to 5, however, we were able to modify our problem and dataset to accurately predict a review in a shorter range.

Limitation:

1. We couldn't find customer reviews of Chinese restaurants dataset in BC.
2. The reviews datasets that were used for statistical tests were integers from 1 to 5, floats data could be more beneficial to get more accurate results.
3. Data from some cities were missing in the Yelp dataset.

Project Experience Summary:

Xu Zhicheng (Max):

1. Cleaning the dataset by removing the missing values and convert the categorical variables into numerical variables.
2. Combine the OSM dataset with Tim Hortons dataset by calculating the distance between them and dropping unnecessary columns.
3. Extracting and processing the features from different columns and use them to give a better prediction on the similarity between two Tim Hortons.
4. Apply KNN and KMEAN to the dataset to get the prediction on the similar Tim Horton.

Rong Li:

1. Cleaned and transformed dataset found online to fit the needs.
2. Interacted with the Pandas library to process and analyze data.
3. Used Post Hoc Analysis to determine if there are mean differences between targets.
4. Used Linear Regression model and Random Forest Classifier to train and predict outcomes given certain conditions.
5. Made revisions to the proposed question so that the select models could produce better predictions while not losing the main focus of the original question.

Zirui Huang

1. Clean, process data and integrate multiple datasets into more organized and connected form.
2. Collect data with external API.
3. Present visualization of prediction and statistical results to audience with seaborn and Plotly library.
4. Apply machine learning tools from scikit-learn to cluster locations. This includes model training and model optimization.
5. Apply NLP tools and machine learning models on textual data. This includes textual data pre-processing, feature extraction with CountVector, and model training.
6. Apply NLP tools like sentiment intensity analyzer to do sentimental analysis.
7. Do statistic analysis to explore relationship between variables.