Final Project
OSM, Photos, and Tours

CMPT353 Fall 2020

Chonghao Huang

Yixu Ye

Overview (Project Experience Summary)

Chonghao Huang

- Analyzed and chose the problems, decided on the solutions and wrote most of the report.
- Used pandas, numpy and other data science skills to find the chain and non-chain restaurants from the amenities.
- Made use of the Mann-Whitney U-Test to do the stats analysis on the chain and non-chain restaurants.
- Filtered the interesting amenities from all the amenities and calculated the distances.
- Did the chi-squared test and the T-test on the Airbnb and amenities data to draw reasonable conclusions.

Yixu Ye

- Collected and took the image data from the cellphone.
- Wrote the code in read_image.py to deal with the images and made the visualization.
- Collected the Airbnb data from online.
- Finished the airbnb_data.ipynb to handle the airbnb dataset and added Machine learning to predict.
- Completed the linear regression for the Airbnb and amenities data.

Problems: These are the problems we tried to answer in this project.

1. Take a collection of geotagged photos representing my walk/tour/vacation, give me a tour of the things I should have seen, or try to guess what is in the photos.
2. Is it true that there are some parts of the city with more chain restaurants (e.g. McDonald's or White Spot franchises, not independently-owned places)? Is there some way to find the chain places automatically and visualize their density relative to non-chains?
3. If I was going to choose a hotel (or AirBnb), where should it be? What places have good amenities nearby?
4. Does the number of customers at an Airbnb place have anything to do with the number of amenities near the Airbnb place?

We analyzed the problem in detail in the following sections, explained the data we collected, talked about the solutions we had, described the techniques we used and showed our results with visualization, numbers or words.

Problem 1, nearby amenities.

Take a collection of geotagged photos representing my walk/tour/vacation, give me a tour of the things I should have seen, or try to guess what is in the photos.

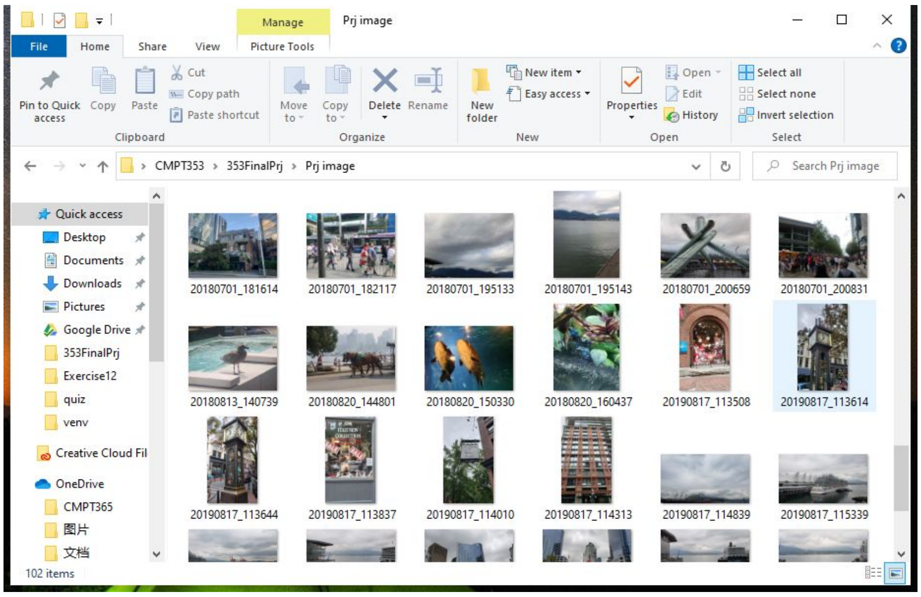
The code for this problem is in the jupyter notebook “nearby_amenities.ipynb”.

For the data, the first thing we did was to filter the amenities data, keep the amenities that we are interested in and filter out the “boring” data points.

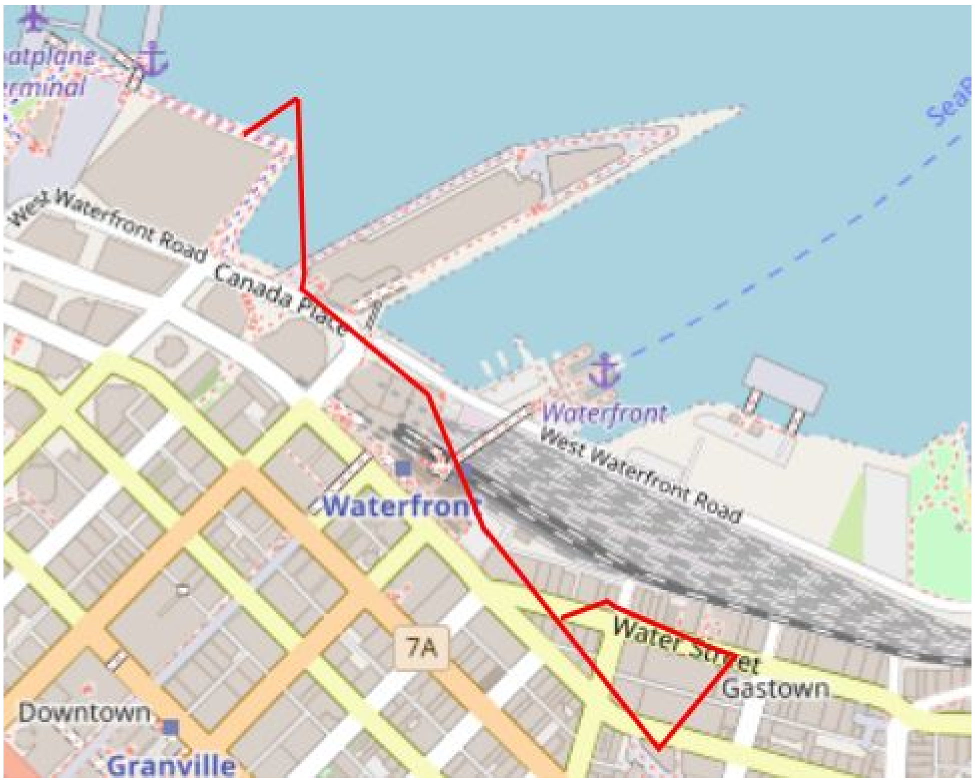
We checked the amenity type and tags in the amenities data. For amenities that have the “tourism” tag, we think they are probably data points that we are interested in because the solution is meant for a tour/vacation, so we kept those data points. For data points that are common amenities, such as “benches” and “waste baskets”, we removed them because they are too common to be seen and do not really mean much for us.

We saved the amenities that we were interested in as a csv file “interesting-amenities.csv” to be used later.

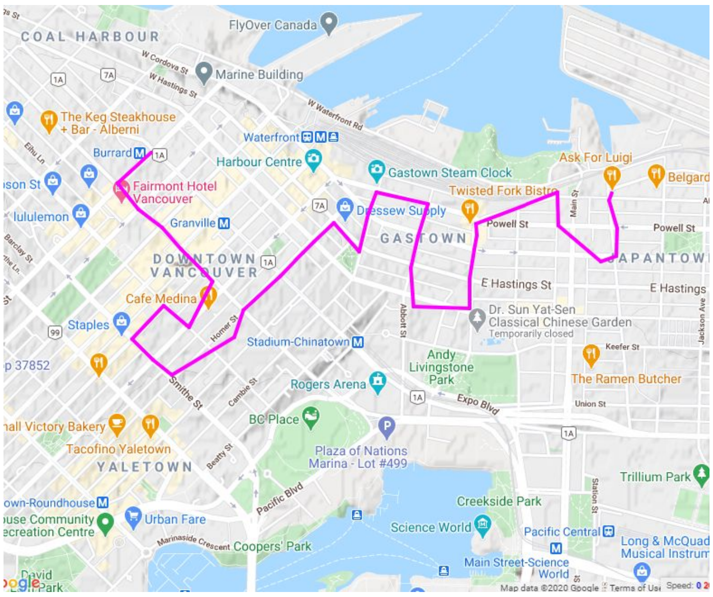
We used some photos from before, the photos were not included in the Git repository because they were too large (>400M), a screenshot of the photos is shown below.



We then used the read_image.py file to generate a csv file of the path. You can see the path in the picture below.



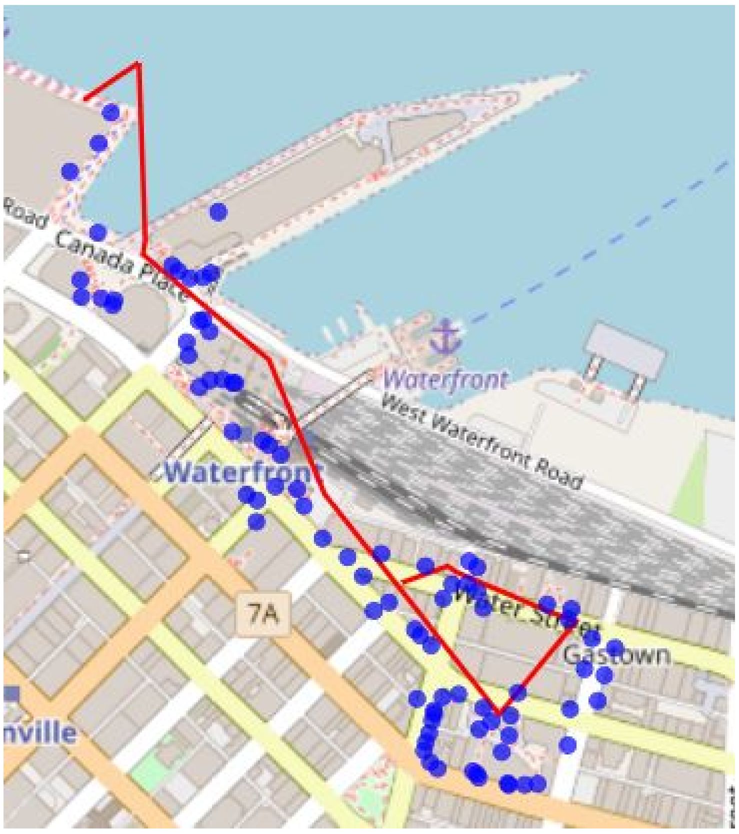
Because the data from the photos were not perfectly accurate, we also had dozens of locations data points representing another walk/tour. So we have two data sets or two paths, one from the photo, one generated by us. These points were generated according to our memories of the last time we took a walk in downtown Vancouver. We decided to use these points as another path and we wanted to find the things that we should have seen along the path. You can see the path in the picture below.



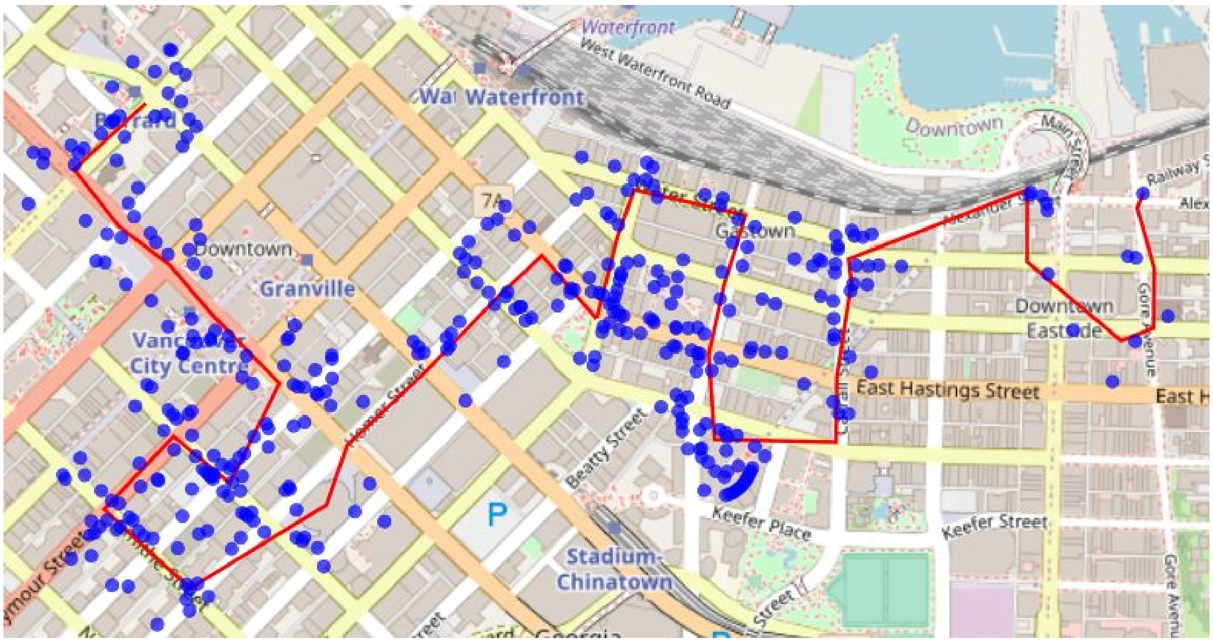
We wanted to generate the data points in a way that the distances are short between one point and the next point, which will make finding the nearby amenities easier. So, we generated one data point at every intersection along the path. The locations are saved in the path.csv file.

Because points in the path are close to each other, we only needed to find the nearby amenities to each point in the path to get all the nearby amenities along the path. To find the nearby amenities, we used the Haversine formula. For every amenity, find the closest point in the path. If it is within 100 m, then this amenity is considered an amenity that we should have seen along the path.

We will show the photo path first. The amenities that we should have seen along the photo path are shown in the picture below.



Now we will show the path generated by us. The amenities that we should have seen along the generated path are shown in the picture below.



We also have a list of the nearby amenities, which includes restaurants, movie theatres and other interesting places. First rows of the list are shown below.

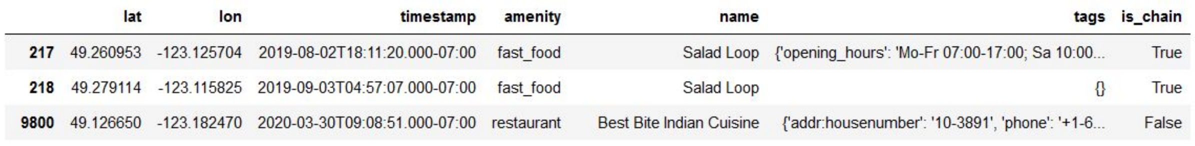
	lat	lon	timestamp	amenity	name	tags
16	49.283192	-123.109050	2015-12-18T21:41:07.000-08:00	pub	The Cambie	{'toilets:wheelchair': 'no', 'wheelchair': 'li...
74	49.280504	-123.106872	2019-12-01T04:03:46.000-08:00	theatre	Cineplex Odeon International Village Cinemas	{'addr:housenumber': '88', 'alt_name': 'Tinsel...
304	49.283265	-123.097896	2019-09-13T13:57:03.000-07:00	restaurant	St Lawrence Restaurant	{'addr:housenumber': '269', 'addr:street': 'Po...
305	49.283288	-123.098074	2019-09-13T13:57:03.000-07:00	restaurant	Cuhillo	{'addr:housenumber': '261', 'addr:street': 'Po...
311	49.283164	-123.103531	2019-09-13T13:57:04.000-07:00	restaurant	The Sardine Can	{'addr:housenumber': '26', 'addr:street': 'Pow...
...	...	...	...	...	...	...

This is the first problem we answered in this project. There are also some limitations to our solution. For example, the data we collect from the photo was not 100% accurate because of the device we were using.

Problem 2, chain restaurants.

Is it true that there are some parts of the city with more chain restaurants (e.g. McDonald's or White Spot franchises, not independently-owned places)? Is there some way to find the chain places automatically and visualize their density relative to non-chains?

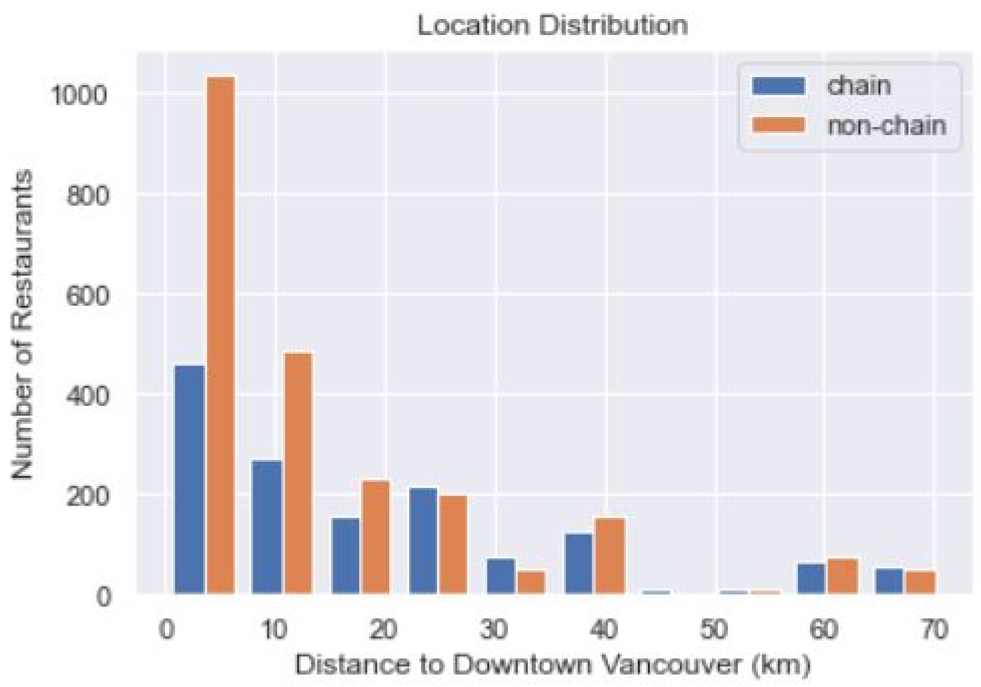
To answer this question, we first tried to find out all the chain restaurants in the amenities data. We kept all the amenities whose types are "fast_food" or "restaurant", grouped them by their names and counted them. If a fast food place / restaurant has a count larger than 1, we consider it a chain restaurant. If it has a count of 1, we consider it an independent restaurant or a non-chain restaurant. This is the method we used to find the chain restaurants, and you can see how we mark them as chain or non-chain in the picture below.



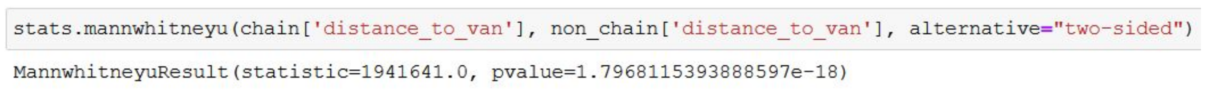
	lat	lon	timestamp	amenity	name	tags	is_chain
217	49.260953	-123.125704	2019-08-02T18:11:20.000-07:00	fast_food	Salad Loop	{'opening_hours': 'Mo-Fr 07:00-17:00; Sa 10:00...	True
218	49.279114	-123.115825	2019-09-03T04:57:07.000-07:00	fast_food	Salad Loop	{}	True
9800	49.126650	-123.182470	2020-03-30T09:08:51.000-07:00	restaurant	Best Bite Indian Cuisine	{'addr:housenumber': '10-3891', 'phone': '+1-6...	False

We decided to answer the question by finding out whether the restaurant's distance to the downtown of Vancouver has anything to do with the chain/non-chain characteristic of the restaurant.

To find out how these restaurants are located, we used the coordinates (49.282875, -123.120464), the location of the Vancouver Art Gallery, to represent the location of the downtown of Vancouver. And we used the Haversine Formula to calculate the distances between every restaurant and downtown Vancouver. We visualized the distribution of the locations of both types of restaurants. The plot is shown below.



As we can see, there is obviously a difference between the distribution of the chain restaurants and the non-chain restaurants. We wanted to prove the conclusion of our observation, so we used the Mann-Whitney U-test to see if we were right. We used an Alpha value of 0.05, and the test showed a p-value of 1.7968115393888597e-18. The result of the Mann-Whitney U-test is shown below.



```
stats.mannwhitneyu(chain['distance_to_van'], non_chain['distance_to_van'], alternative="two-sided")
MannwhitneyuResult(statistic=1941641.0, pvalue=1.7968115393888597e-18)
```

The p-value is way smaller than 0.05, so we think that there is a difference between the chain and non-chain restaurants. Looking at the plot, we think that non-chain restaurants tend to be nearer to the downtown, and chain restaurants tend to be more spread out instead of all gathering near the downtown. Within 10 kms of the downtown, there are a lot more non-chain restaurants than chain restaurants. But if you go deeper into the suburbs, for example, when you are more than 20 kms from the downtown, you will probably see as many chain restaurants as non-chain restaurants, or even more.

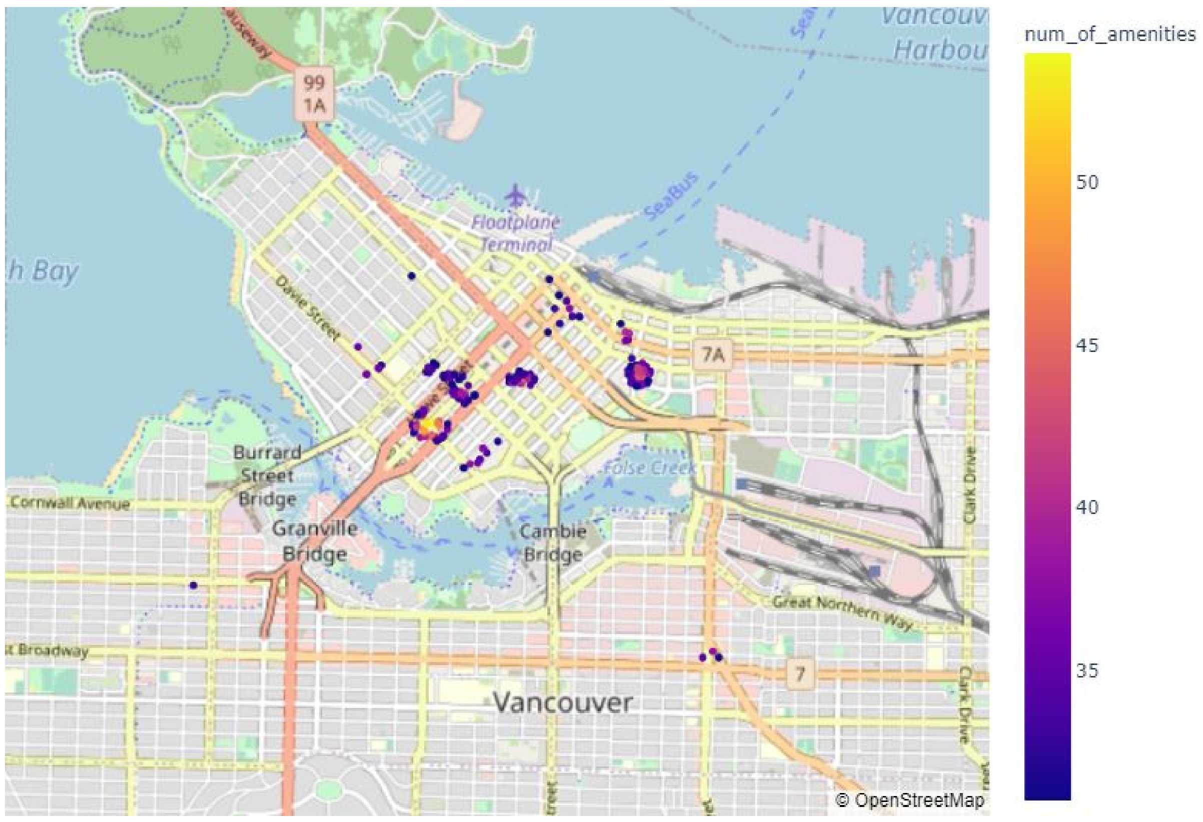
For this problem, we used the Mann-Whitney U-test as our method. There were also some limitations to our solution. For example, the Mann-Whitney U-test has its limitations. We wanted to do a T-test, but we could not transform the data to the best shape.

Problem 3, choose Airbnb with most amenities.

If I was going to choose a hotel (or AirBnb), where should it be? What places have good amenities nearby?

To answer this question, we first got the data of all the Airbnb places in Vancouver from http://insideairbnb.com/get-the-data.html. We also needed the data of good amenities, and we had the interesting_amenities data from previous problems so we used it here. And then we calculated the number of amenities nearby for every Airbnb place using the Haversine Formula.

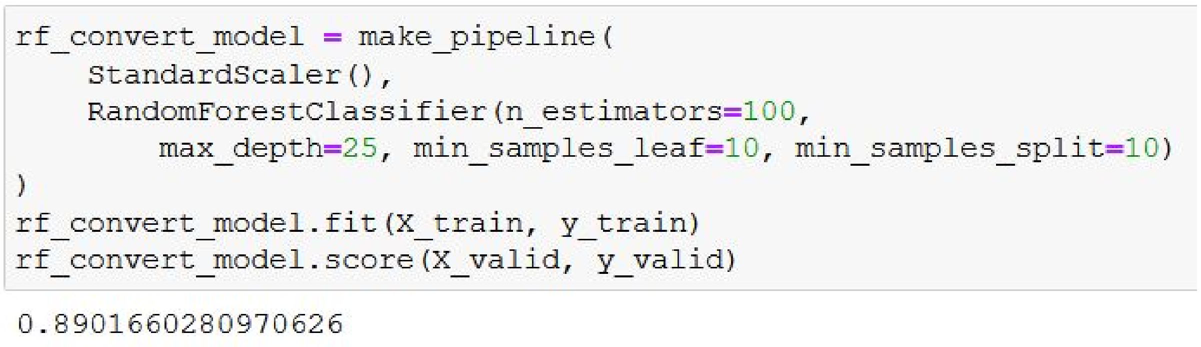
Because we want to know which Airbnb places have good amenities nearby, we found all the Airbnb places that have more than 30 good amenities nearby, and plotted them on the map. The result is shown below.



As we can see, these Airbnb places gather in several areas, including the International Village, Yaletown and Robson Street. This is one way to answer the question because we have a list of the Airbnb places that have the most good amenities nearby and we have also shown you where these areas are, so even if you do not like the list we give you, you can always go find some other hotels or Airbnb places in these particular areas.

To answer the question in another way, we also did something else. We tried to make a model to find the Airbnb places that have good amenities near them. In order to do this, we used machine learning.

We used two columns in the data as our X, which are “lat” and “lon”. And “has_amenities” as our y. We used a random forest classifier as our model and we reached a score of 0.8901660280970626. The result is shown below.



```
rf_convert_model = make_pipeline(
    StandardScaler(),
    RandomForestClassifier(n_estimators=100,
        max_depth=25, min_samples_leaf=10, min_samples_split=10)
)
rf_convert_model.fit(X_train, y_train)
rf_convert_model.score(X_valid, y_valid)
```

```
0.8901660280970626
```

We believe our model does a great job at finding the Airbnb places with good amenities nearby. There were also some limitations to our solution. For example, although the accuracy is good and we are satisfied, it is still less than 0.9.

Problem 4, Airbnb popularity and amenities.

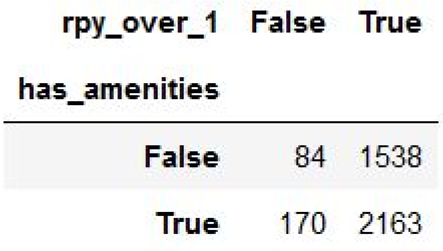
Does the number of customers at an Airbnb place have anything to do with the number of amenities near the Airbnb place?

1. The chi-squared test.

To answer this question, we wanted some data that can represent the number of customers at an Airbnb place, but we did not have the “number of customers per month” data or anything similar. However, we did have the data of the number of “reviews per month”, which we believe can represent the number of customers at the place, so we decided to use that data.

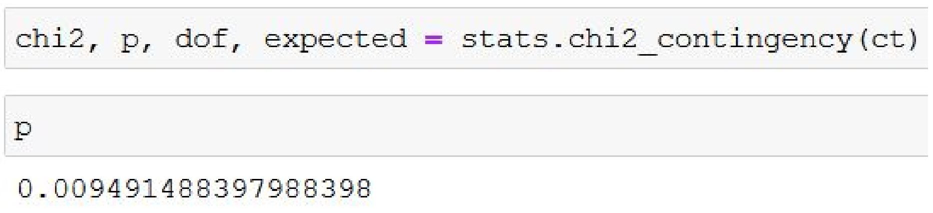
We already have the “number of amenities nearby” data from previous problems so we will use it here.

We decided to do a chi-squared test. First, we divided the Airbnb places into two categories, one has at least 1 review per year, the other has less than 1 review per year. And second, we also divided the Airbnb places into another two categories, one has at least 1 amenity within 100m of it, the other has no amenities within 100m of it. We made a contingency table. The result is shown below.



rpy_over_1	False	True
has_amenities		
False	84	1538
True	170	2163

As we can see in the table, it looks like places with no amenities nearby tend to have less rpy_over_1. We wanted to check if this observation is correct, so we did the test, and the p-value was 0.009491488397988398, which is smaller than 0.05. The result is shown below.



```
chi2, p, dof, expected = stats.chi2_contingency(ct)
```

```
p
```

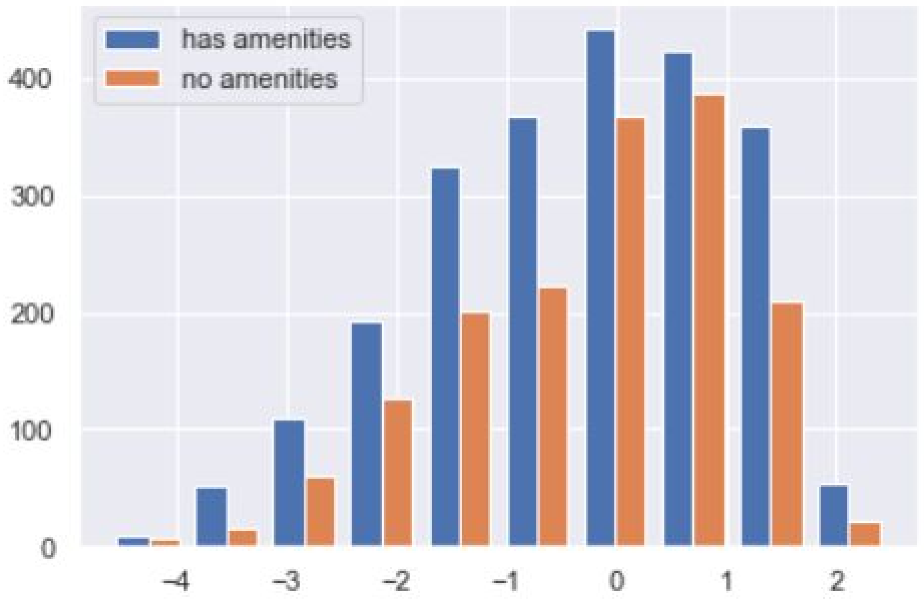
```
0.009491488397988398
```

So, we concluded that our observation was correct: whether the place has amenities nearby has some effect on whether they get more than 1 review per year. And we think that places with no amenities nearby tend to have less rpy_over_1.

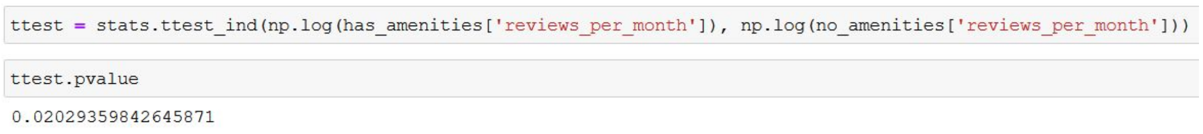
2. The T-test.

We also tried to approach this problem using another method. We split the data into the two categories. One has at least 1 amenity within 100m of it, the other has no amenities within 100m of it. And we tried to do a t-test to see if their "reviews_per_month" data are different.

We transformed the data using np.log and the result looks normal. The result is shown below.



Also, we have thousands of data points, which is much more than 40. So, we decided that the data was ready for a t-test. We did the t-test and got a p-value of 0.02029359842645871, which is less than 0.05. The result is shown below.



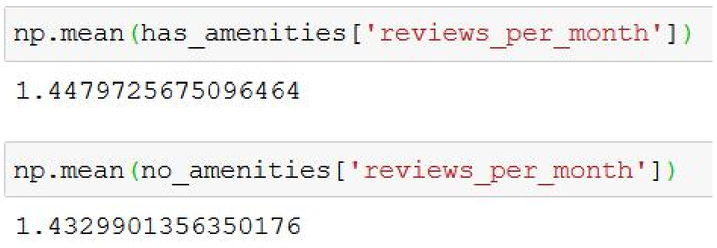
```
ttest = stats.ttest_ind(np.log(has_amenities['reviews_per_month']), np.log(no_amenities['reviews_per_month']))
```

```
ttest.pvalue
```

```
0.02029359842645871
```

Because the p-value is less than 0.05, our Alpha value, we concluded that Airbnb places that have amenities nearby and places that have no amenities nearby have different numbers of reviews per month.

We also computed the average reviews_per_month for both groups, and the result was that those with amenities nearby have a higher average. The result is shown below.



```
np.mean(has_amenities['reviews_per_month'])
```

```
1.4479725675096464
```

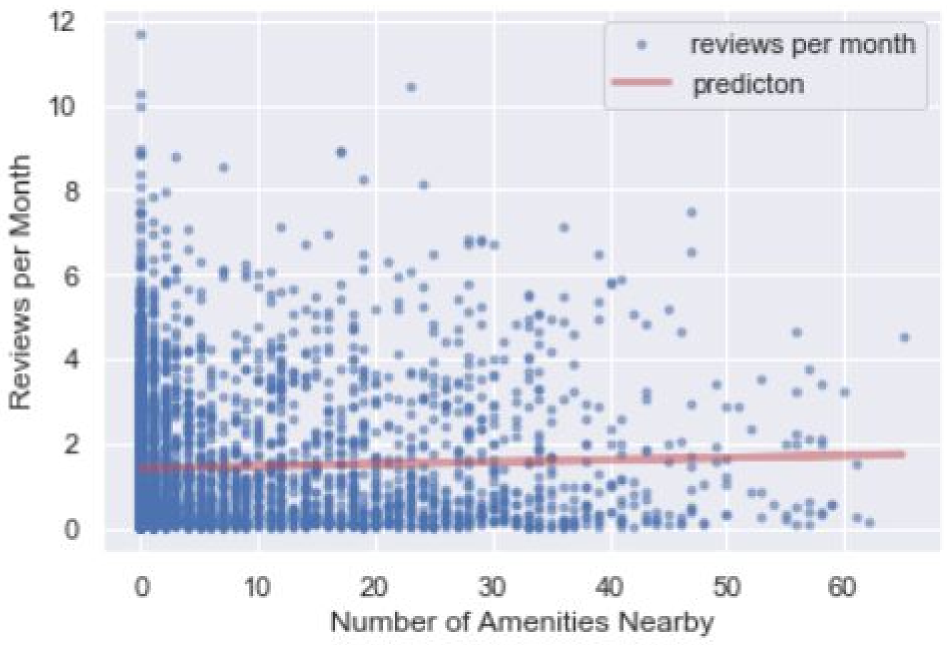
```
np.mean(no_amenities['reviews_per_month'])
```

```
1.4329901356350176
```

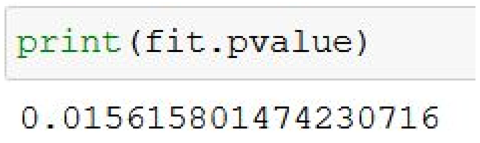
So, we think Airbnb places that have amenities nearby have more reviews per month.

3. The regression

We also did a linear regression to further support our opinion. We did a linear regression using the number of amenities nearby and the number of reviews per month that each Airbnb place has. The result is shown below.



The slope is 0.005014368875876933, and the intercept is 1.4039141120444443. Judging from the plot, the number of reviews per month gets higher when the place has more amenities nearby. To prove this, we check the p-value of the regression. It is 0.015615801474230716, which is less than 0.05, and it is shown below.



```
print(fit.pvalue)
```

```
0.015615801474230716
```

We think this further supports the conclusion that Airbnb places tend to get more reviews when they have more amenities nearby. There were also some limitations to our solution as well. For example, it is worth noting that the residuals of the linear regression did not look perfect, although we do not think it makes our conclusion incorrect.