

Clustering of Restaurants from Vancouver and Seoul

k- means Clustering

17101936 Kim NamYeong

20102104 Kim Sua

22170091 Lea Scharpf

Business Analytics | 29.11.2022

# Introduction

Opening a restaurant in big capital cities can be quite a challenge. Not only good food needs to be saved in order to not go bankrupt, but also the people living there should value the food. To find the best location to open a restaurant depends on a lot of different factors. People in that area should earn enough money to be able to go to a restaurant. Next, if you want to open a restaurant with a cuisine from another country, some people from that country should live close to the restaurant, to be sure to have some customers. Also, it makes a restaurant more authentic, when people from that country enjoy the food. Third, there should not be too many competitors, since people than have to choose between your restaurant and others.

Therefore, we choose to cluster the restaurants from Vancouver and Seoul in order to find the best location to open a restaurant from a specific type of cuisine. The goal is to reduce business failures due to incorrect market analysis.

# Data import

To create such a clustering method, datasets from both cities including income, citizenship and a restaurant dataset are needed.

Kaggle provided the restaurant dataset for Vancouver:

* <https://www.kaggle.com/datasets/banaveenkumar/vancouver-restaurent-dataset>

The dataset for the income and the citizenship was found at Opendata Vancouver:

* <https://opendata.vancouver.ca/explore/?refine.theme=Demographics&disjunctive.features&disjunctive.theme&disjunctive.keyword&disjunctive.data-owner&disjunctive.data-team&sort=modified>

For the city Seoul the datasets were available on data Seoul:

* Korean dataset of restaurant:
  + <https://data.seoul.go.kr/dataList/OA-16094/S/1/datasetView.do?tab=S>
* Korean dataset of foreigner:
  + <https://data.seoul.go.kr/dataList/803/S/2/datasetView.do?tab=S>

# Preprocessing

After importing these data sets it was necessary to do preprocessing steps to clean the datasets.

## Vancouver dataset

Since the Vancouver census datasets consists of a lot of different spreadsheets, it was necessary to extract the important information. Therefore, two new datasets were created out of the census dataset only containing the necessary information.

Before preprocessing the datasets looked like this:



Figure 1- Dataset of citizenship before pre-processing

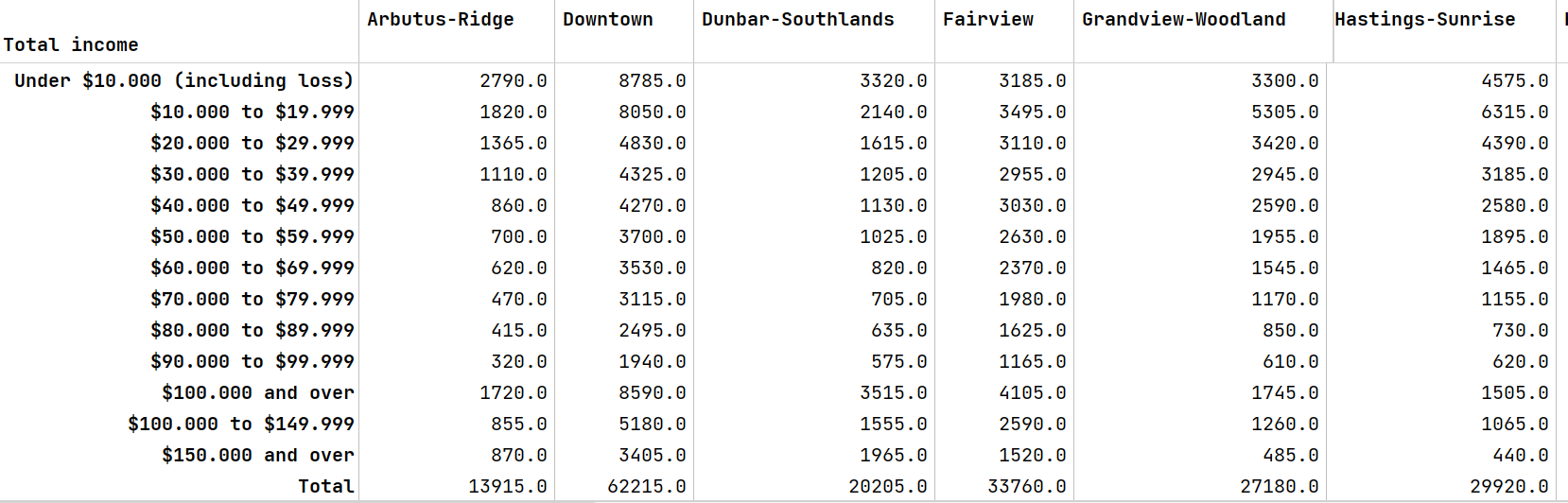


Figure 2- Dataset of income before pre-processing

* + 1. Pre-processing of Vancouver income dataset

The Vancouver income dataset contains the features: total income and the districts of Vancouver. 24 districts of Vancouver are included. Every instance is about the range of income, going from under 10 000$ in 12 steps up to 150 000$ income and more.

The row “$100.000 and over” needs to be dropped because it only contains the information found in the rows before and after.

We are only interested in the weighted average income from each district. Therefore, it is necessary to calculate the average income first and then calculate the total weighted average income for every district.

After applying these calculations, the dataset looked like this:

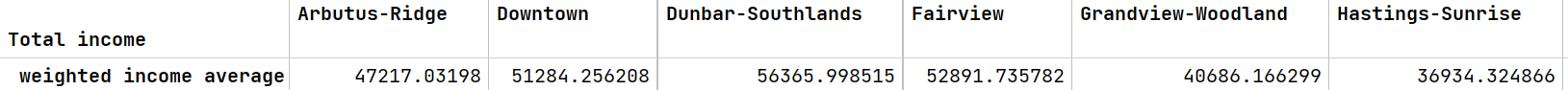


Figure 3- Vancouver dataset for income after pre-processing

* + 1. Pre-processing of Vancouver citizenship dataset

The citizenship dataset from Vancouver includes features of the place of birth and number of people of that birthplace in every district.

First, the ID column had to be dropped.

Then, I transposed the dataset to better calculate the average population of each district. Since we decided to do the clustering only for the categories: Western, Japanese, Koreans, Chinese, South- East Asians, I summarized these origins, calculated the total population and the percentage of the population in every district and dropped the rest. As Western people I included people from Europe, the USA and Canada. After the preprocessing the dataset looked like this:

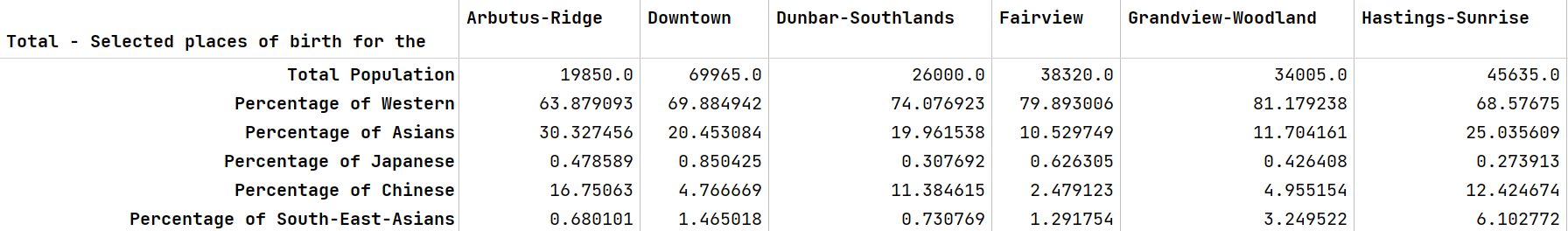
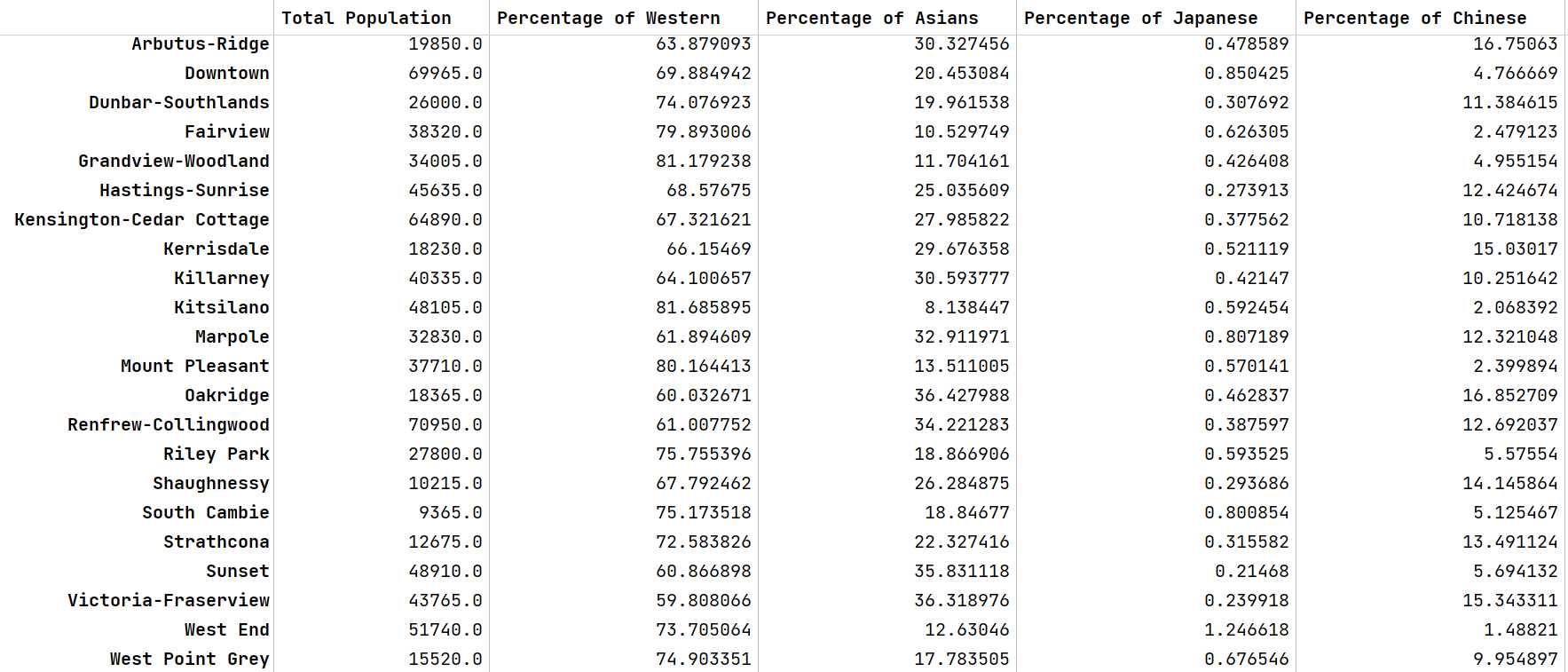
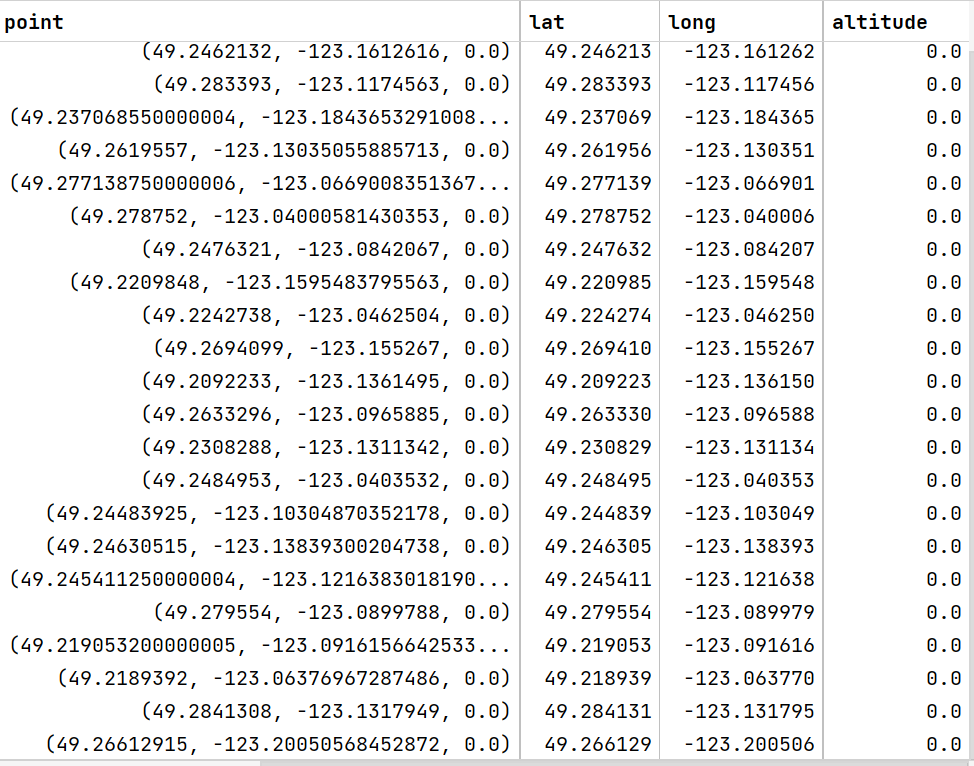


Figure 4- Vancouver citizenship after preprocessing

These two datasets are now ready to be merged.

To get a better overview of the districts of Vancouver a folium map was created with the help of Geopy. Geopy provides a class for popular mapping services. Nominatim is the service behind the popular OpenStreetMap that allows you to geocode for free. With geocode it is possible to get the longitude and latitude of a place. After merging and calculation the location the final dataset looked like this:



With Folium Map it is possible to create a map displaying every district:

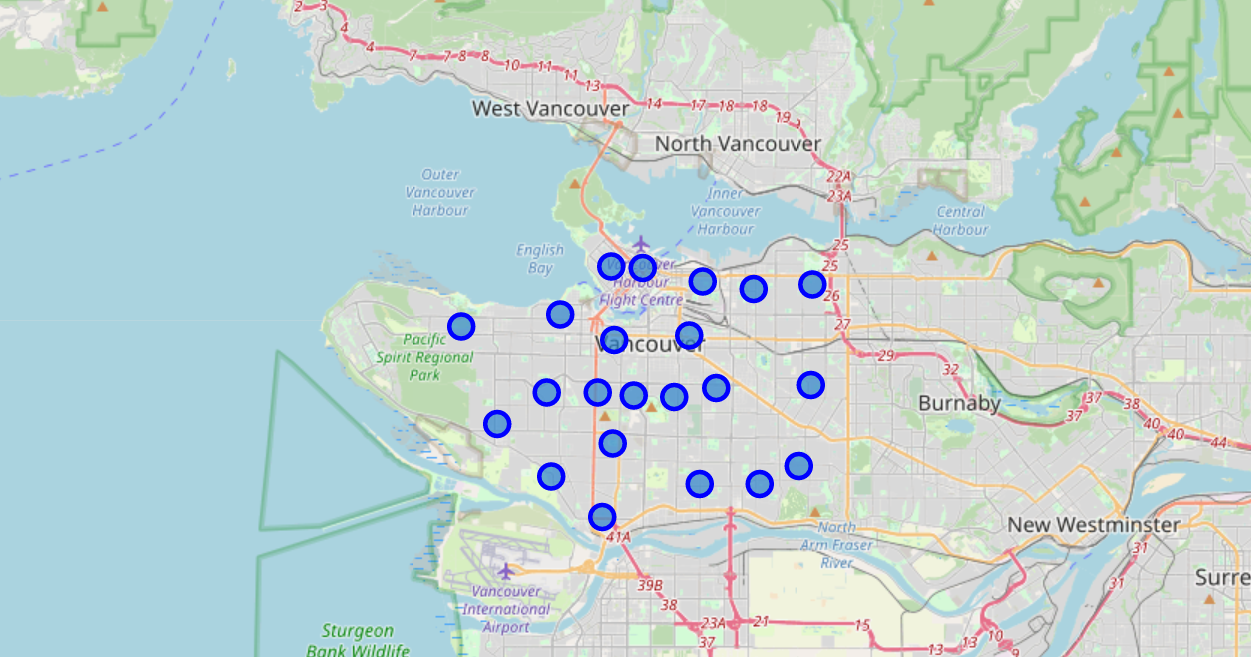
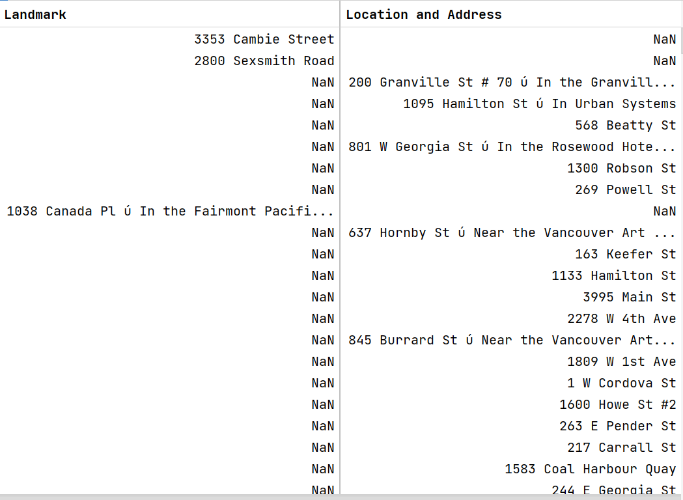


Figure 5- Folium Map of Vancouver districts

* + 1. Preprocessing of Vancouver restaurant dataset

The dataset provided from Kaggle contains the features: name of restaurant, type of cuisine, rating, total no of ratings, cost, landmark, opening time, current status, dine in availability, takeaway type, delivery availability and location and address. From this given features only the features: name of restaurant, type of cuisine, landmark and location and address are interesting for the given task.

In total only 384 instances are given, which makes this dataset comparably small.

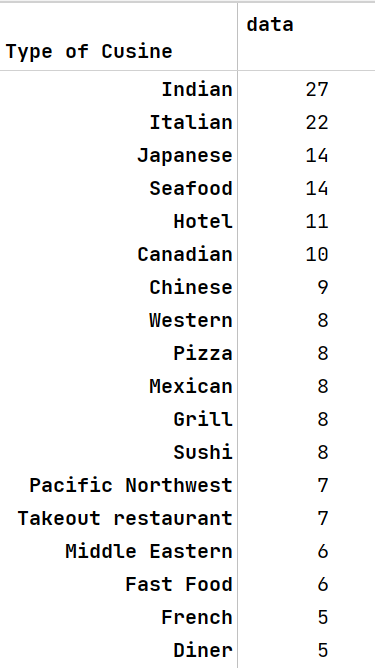
After dropping the unnecessary features, the dataset looked like this:

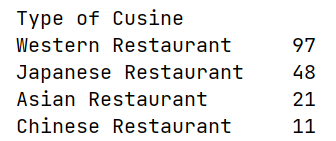
The dataset contains two duplicates, which need to be dropped at first.

Having a closer look on the instances, one can see the challenges this dataset provides:

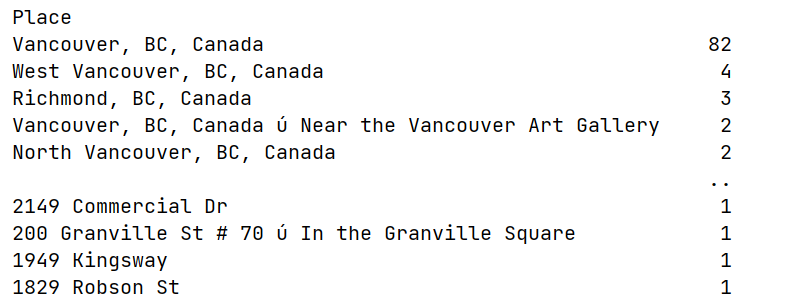
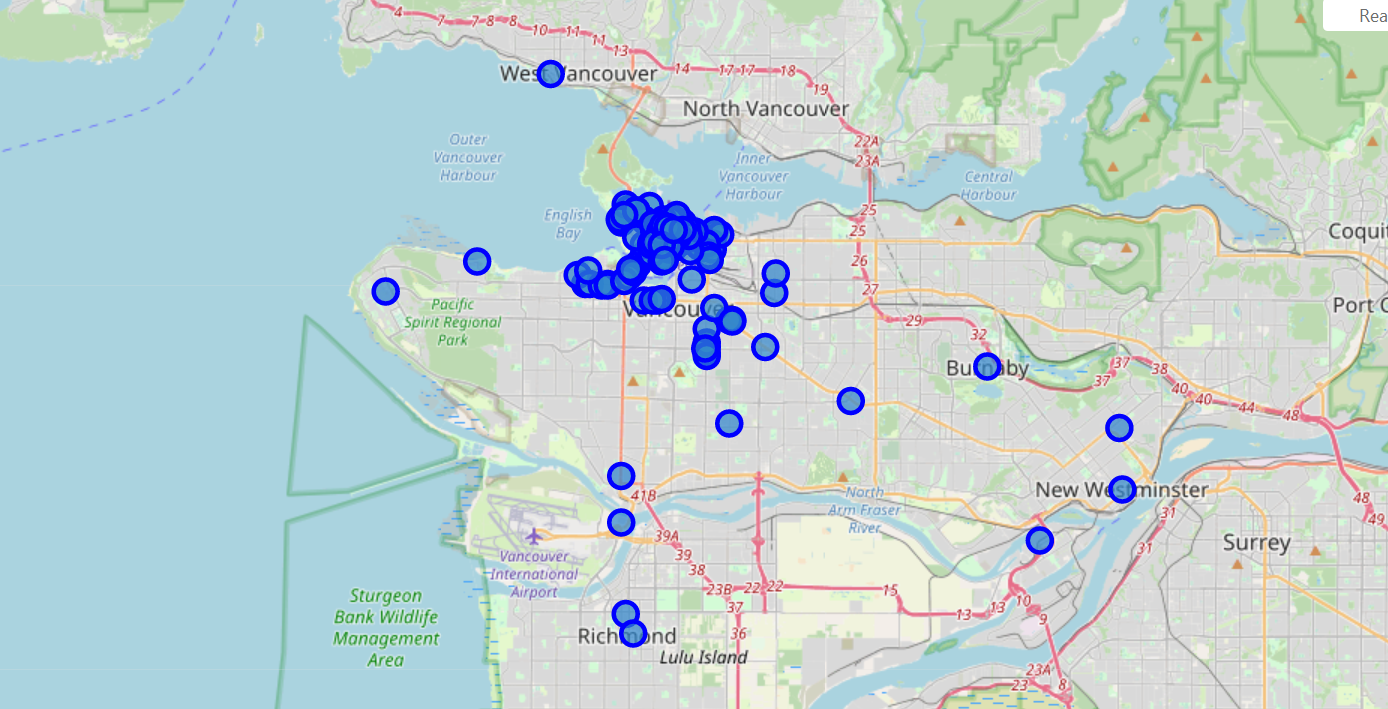
1. The type of cuisine is often referred to as “Restaurant”, which has no meaning at all.
2. The Landmark and Location and Address contain a lot of NaN values.

Since the dataset is so small it is not possible to just drop the 73 rows that contain as type of cuisine only “Restaurant”. Therefore, it was necessary to extract the information about the type of cuisine from the restaurant’s name. After doing so from the 382 instances 325 instances were left.

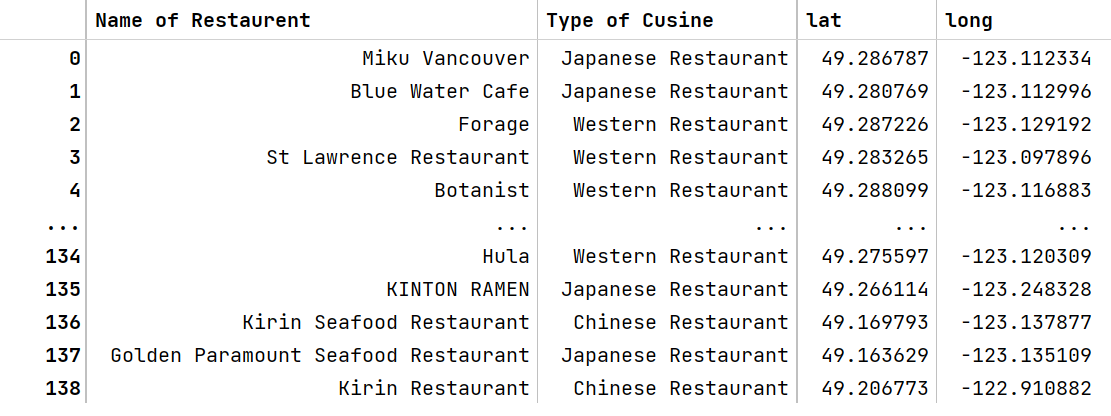
Our goal was to cluster the districts with the types of restaurants: Western, Chinese, Japanese, Rest of Asia. That’s why these types of cuisines need to be aggregated to the categories mentioned before. It resulted in the following number of types of cuisine:



Regrettably only 177 instances are left.

The second challenge was to find the exact location of the restaurant. I merged the Landmark and Location and Address since they have the same meaning into a new column “place”. To be able to get the exact location with the help of geopy meaningful addresses need to be provided. Having a look at the type of places leads to a sobering result. 82 instances are located in Vancouver, but do not contain a specific address. I choose to add the name of the restaurant to the first three instances, so geopy might locate the restaurants. Another problem are the addresses containing “#” or “ú”. With some more feature engineering it was possible to get more exact locations. After applying the geopy location function I choose to drop the instances that still couldn’t be found. That resulted in a dataset with 138 instances left. With the help of folium map we could visualize to location of the restaurants.

The final restaurant dataset is visualized in the following figure:



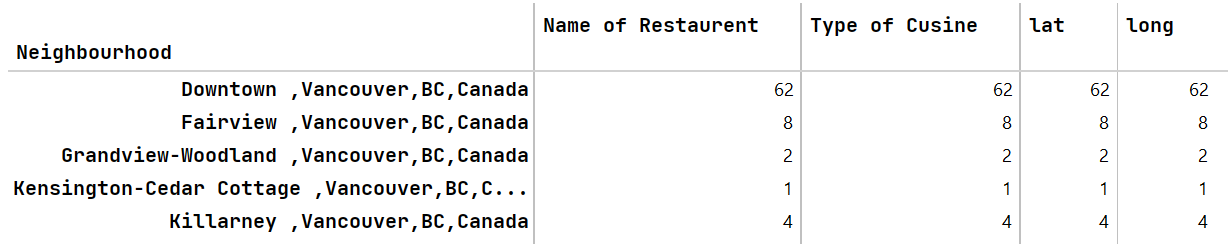
* + 1. Final merge of the datasets

Now, the two resulting datasets need to be merged. The challenge hereby is to merge the datasets on the feature district/ neighborhood. Since the latitude and longitude of the restaurant and the neighborhood are given, I merged the datasets using the Haversine Distance.

“The Haversine formula calculates the shortest distance between two points on a sphere using their latitudes and longitudes measured along the surface.” ([Haversine formula to find distance between two points on a sphere - GeeksforGeeks](https://www.geeksforgeeks.org/haversine-formula-to-find-distance-between-two-points-on-a-sphere/)) Mathematically is can be expressed as: . Calculating the central angle of the haversine and solving it for d results in:

Applying this formula on the longitude of the restaurant and the neighbourhood, it was possible to find the closest neighbourhood for every restaurant.

The restaurants are mostly distributed over Downtown, Vancouver, as the following figure shows.



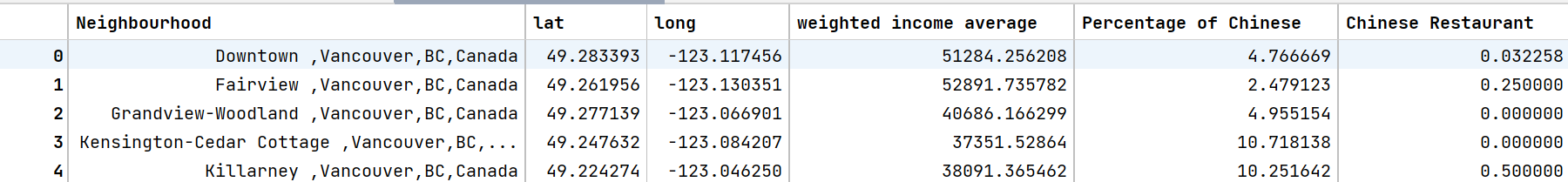
Next step is to use one hot encoding to create numerical values from the four types of cuisine: Chinese, Japanese, Asian and Western Restaurant:



Depending on which restaurant type we want to create the clustering for, the “str\_cluster” needs to be adjusted.

In the following example I will make the analysis for Chinese restaurants. The dataset shown in the previous figure can therefore be adjusted to only the neighbourhood and the type of restaurant we want to do the clustering for.

After merging the datasets on “neighbourhood”, the final dataset is shown in the following figure:



Still, we need to normalize our data. In this case using Standard Scaler.

# Model and Training

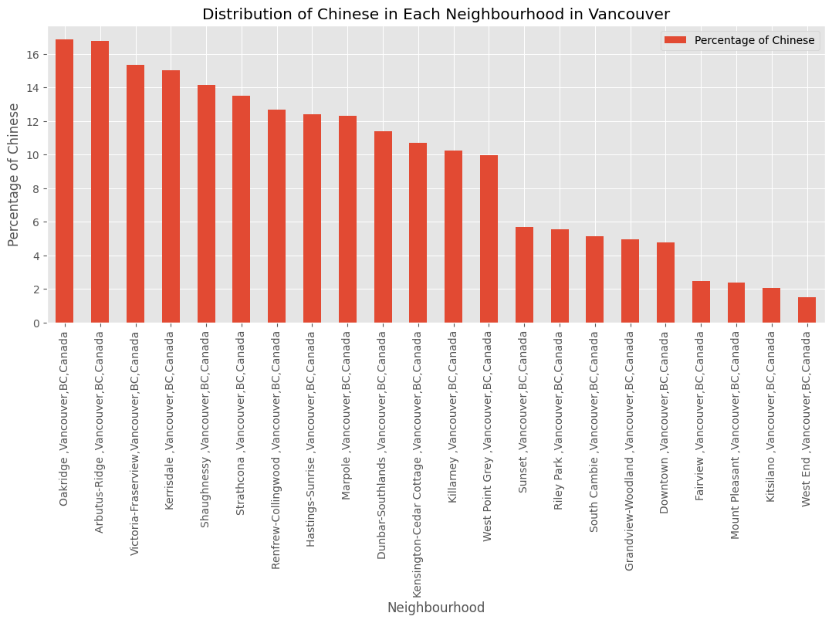
We decided to use the k-means Clustering for our model, since we have an unlabeled dataset and want to find useful properties of the structures of the dataset. Clustering is the task of partitioning the dataset into groups, called clusters. K- means Clustering particularly finds k cluster centers that are representative of certain regions of the data.

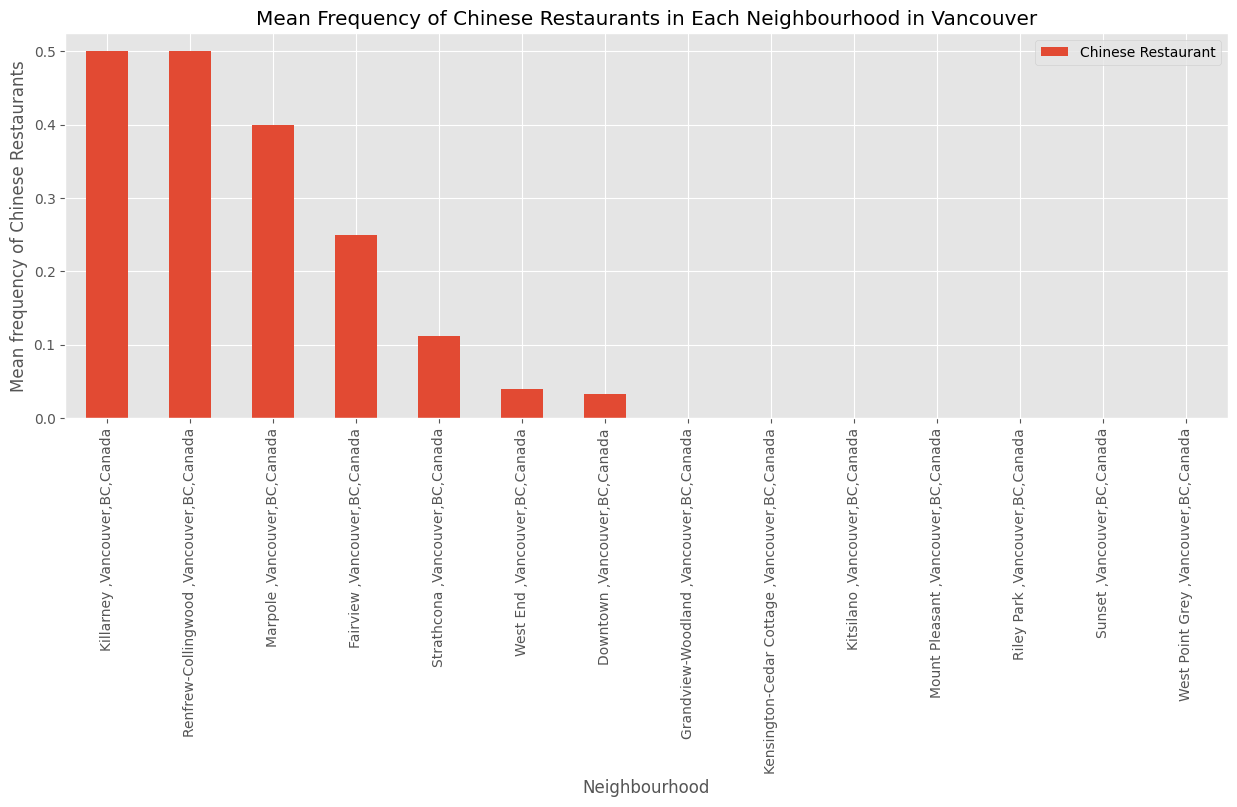
The algorithm starts by selecting kk points as the initial cluster centers randomly and afterwards alternated between the steps: Expectation and Maximization. The result will be displayed, when the assignment of data points to clusters no longer changes.

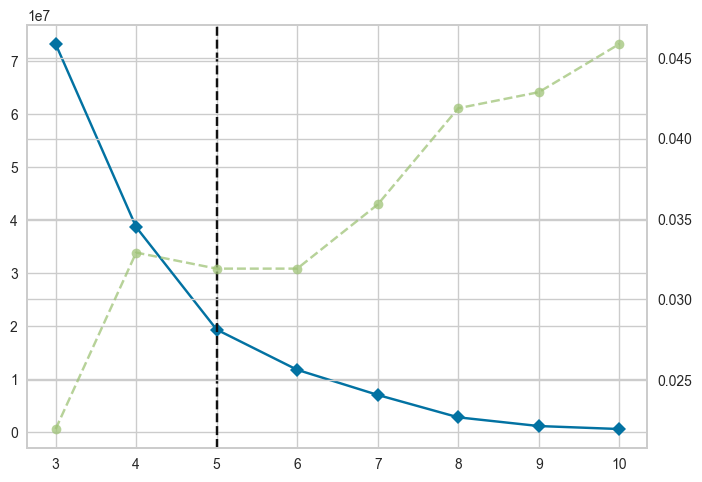
## vancouver dataset

* + 1. Visualization of distribution

To get a better overview of the distribution of the neighborhoods, three plots were created, visualizing the three main features: income, citizenship and restaurants.



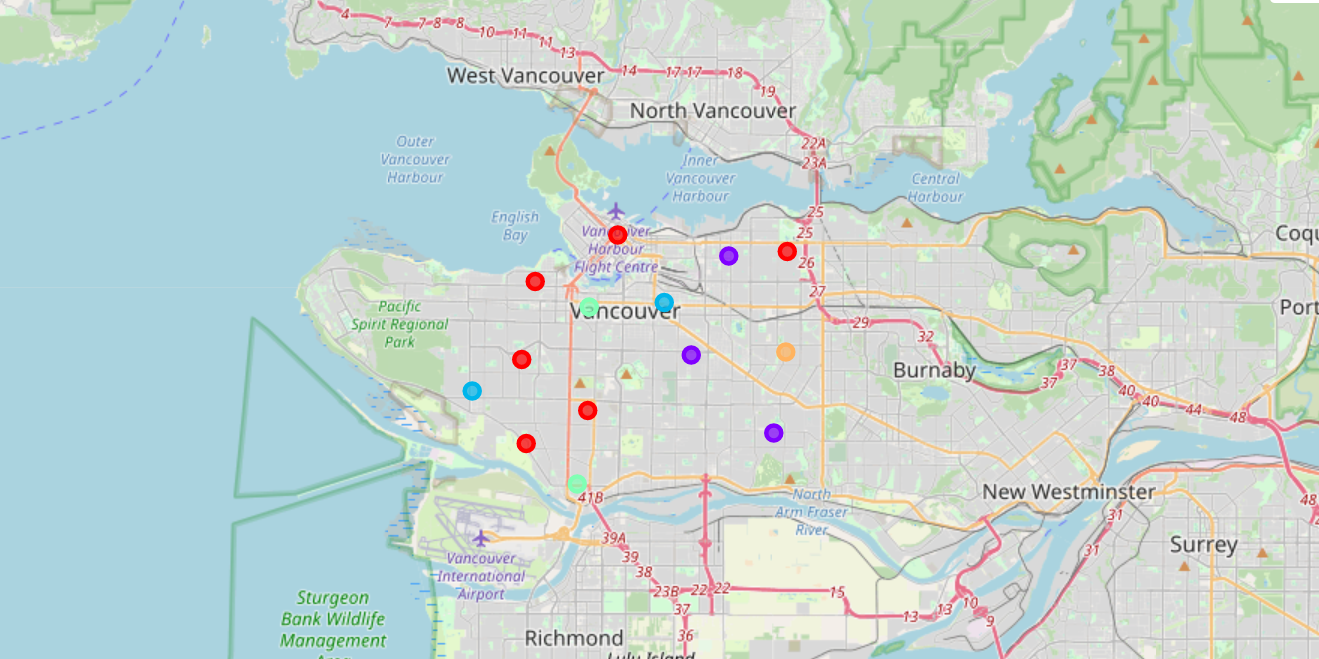


* + 1. K- means clustering

Before we train our model, we need to find out the hyperparameter k. To do so the squared error calculated respectively were used as metrics of their performances. An analysis using K Elbow Visualizer and Squared error for each k value evident shows that k = 5 would the best value.

* + 1. Interpretation of resulting clusters

The k- clustering result for the Chinese restaurants is visualized in the following figure:



The final step is to interpret the five clusters.

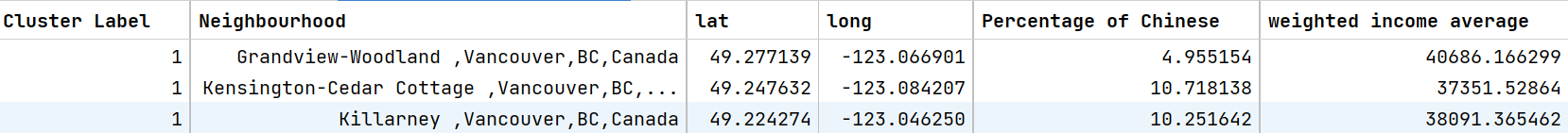
Cluster 0: (red)



* Percentage of Target Customers: HIGH
* Spending Power: MID
* Number of Competitors: LOW

Since the percentage of target customers is high and the number of competitors low, I would recommend opening a restaurant here.

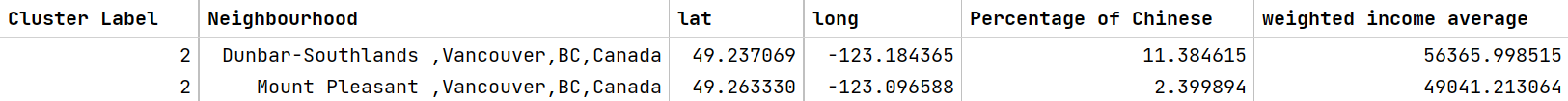
Cluster 1: (purple)



* Percentage of Target Customers: MID
* Spending Power: LOW
* Number of Competitors: HIGH

Since the spending power is low and the number of competitors high, I would not recommend to open a restaurant here.

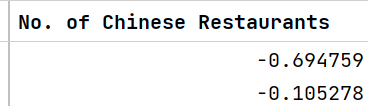
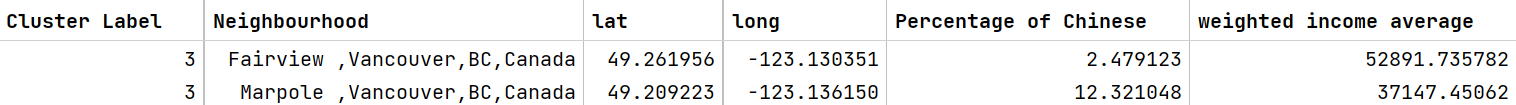
Cluster 2: (blue)



* Percentage of Target Customers: MID
* Spending Power: HIGH
* Number of Competitors: LOW

Cluster two can also be recommended, as there are only a few competitors and high spending power.

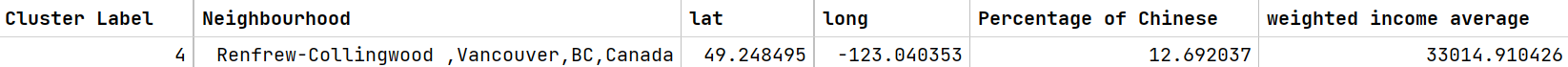
Cluster 3: (turquoise)



* Percentage of Target Customers: MID
* Spending Power: MID
* Number of Competitors: MID

I would not recommend opening a restaurant here, since nothing is special about this cluster.

Cluster 4: (orange)



* Percentage of Target Customers: HIGH
* Spending Power: LOW
* Number of Competitors: LOW

The spending power is low, but there are a lot of potential customers and less competitors. Therefore, this cluster can also be an option.