Applied Data Science Capstone – The Battle of Neighborhoods I

The housing price at Great Toronto Area

By Daniel Chen

➢ Motivation

With the growth of population at Canada, demands of houses has been growing. By doing the research of housing price at great Toronto area will help people who want to invest on estate or buy for living. For example, if there is an investor wants to do the investment for buying a house at Toronto; it is crucial to having the information about what kinds of venue around it. In advance research, the investor can do the prediction of pricing on the specific area. The other advantage benefits to people like me; plan to buy a house that contents personal living requirements. By doing the research on neighborhood at Toronto is helping me to decide where will be the best fit for me to live. In addition, including the budget can assist me to choose the house is affordable and the area is suitable to me.

➢ Data

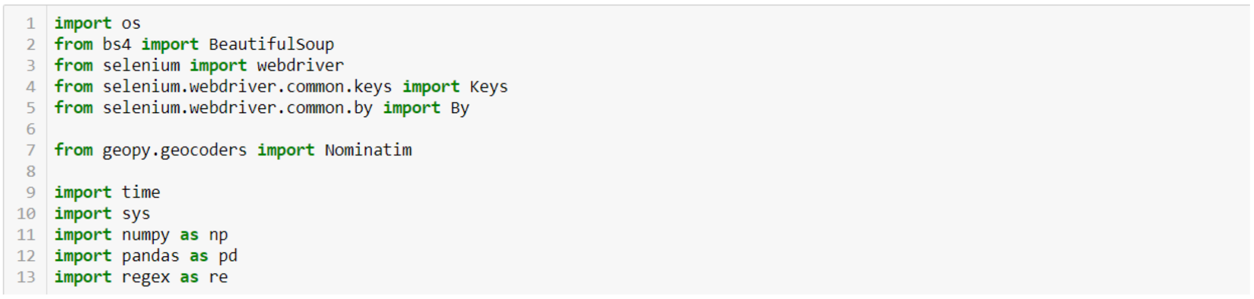
Housing prices **–** ***Zoocasa***

- Scraped by Beautifulsoup

Location – ***Nominatim of GeoPy***

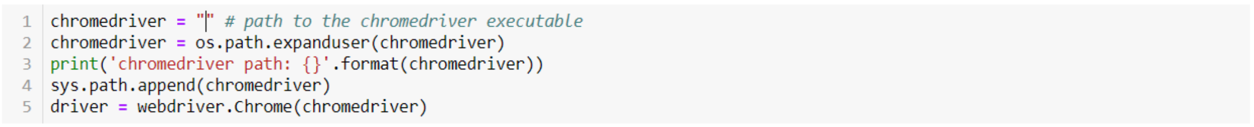
Venues – ***Foursquare***

- **Code** – web scraping housing prices from zoocasa
    - Import libraries



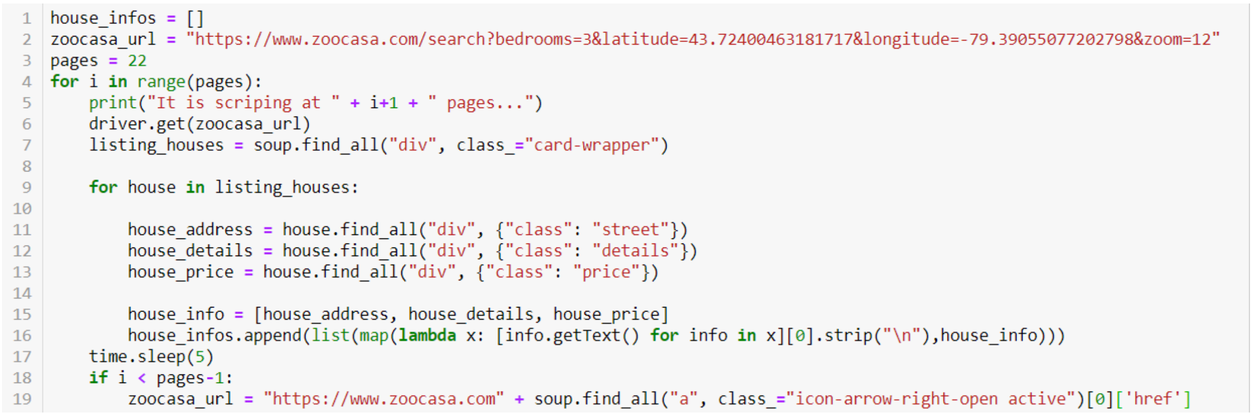
```python
import os
from bs4 import BeautifulSoup
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By

from geopy.geocoders import Nominatim

import time
import sys
import numpy as np
import pandas as pd
import regex as re
```

- Applying Chrome drive



```python
chromedriver = "" # path to the chromedriver executable
chromedriver = os.path.expanduser(chromedriver)
print('chromedriver path: {}'.format(chromedriver))
sys.path.append(chromedriver)
driver = webdriver.Chrome(chromedriver)
```

- Scraping



```python
house_infos = []
zoocasa_url = "https://www.zoocasa.com/search?bedrooms=3&latitude=43.72400463181717&longitude=-79.39055077202798&zoom=12"
pages = 22
for i in range(pages):
    print("It is scriping at " + i+1 + " pages...")
    driver.get(zoocasa_url)
    listing_houses = soup.find_all("div", class_="card-wrapper")

    for house in listing_houses:

        house_address = house.find_all("div", {"class": "street"})
        house_details = house.find_all("div", {"class": "details"})
        house_price = house.find_all("div", {"class": "price"})

        house_info = [house_address, house_details, house_price]
        house_infos.append(list(map(lambda x: [info.getText() for info in x][0].strip("\n"),house_info)))
    time.sleep(5)
    if i < pages-1:
        zoocasa_url = "https://www.zoocasa.com" + soup.find_all("a", class_="icon-arrow-right-open active")[0]['href']
```

- **Making Dataframe** – *Including Address, Home Type, Price, Latitude, and Longitude*
  - Dataframe from Zoocasa – House Postal code, Address, type, and price

```
df = pd.DataFrame(data = house_infos)
df.columns = ['Address', 'Home Type', 'Price']

#Make price value to be integer
df["Price"] = [str(x[2].replace("$","").replace("\n","").replace(",", "").replace(" ","")) for x in house_infos]

#Make adress to be available searching location on Nominatim
addr_series = df["Address"].copy()
for idx, address in enumerate(address_series_latlon):
    if "-" in address:
        minusidx = address.index('-')
        addr_series.set_value(idx, addr_series[idx][minusidx+1::])

#Getting postal code from geocoder.canadapost
postalcodes = []
for address in addr_series:
    g = geocoder.canadapost(address + ", Toronto")
    postalcodes.append(g.postal[0:3])
df.insert(0, "Postal Code", postalcodes)
```

  - Determine the postal code belongs to Toronto

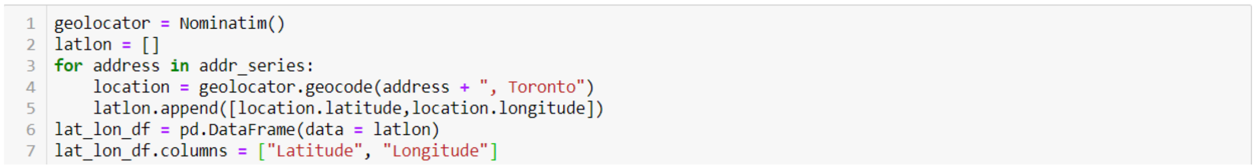
```
!wget -q -O 'Geospatial_Coordinates.csv' http://cocl.us/Geospatial_data
unique_postal = pd.DataFrame({'Postal Code':df["Postal Code"].unique()})

lat_lng_coords = pd.read_csv('Geospatial_Coordinates.csv')
#Delete the postal code not belong to Toronto
postal_notToronto = []
for postalcode in unique_postal['Postal Code']:
    if postalcode not in lat_lng_coords['Postal Code'].tolist():
        unique_postal = unique_postal[unique_postal['Postal Code'] != postalcode].reset_index()
        postal_notToronto.append(postalcode)
unique_postal
```

	index	Postal Code
0	0	M4J
1	1	M1K
2	2	M1P
3	3	M5P
4	4	M4L
5	5	M5J
6	6	M5B
7	7	M1J
8	8	M5T
9	9	M6P
10	10	M1L
11	11	M5S
12	12	M4R
13	13	M6H
14	14	M5V
15	16	M4A

  - Clean the data of house not in Toronto

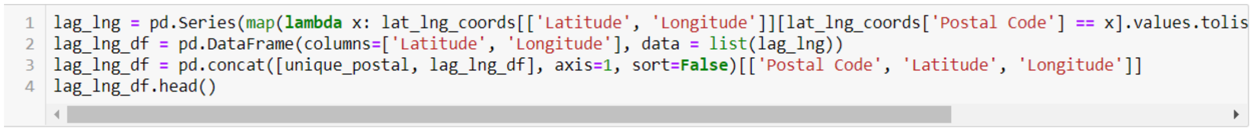
```
for postalcode in postal_notToronto:
    df = df[df['Postal Code'] != postalcode].reset_index()
```

- Dataframe from Nominatim – House Coordinate (latitude and longitude)

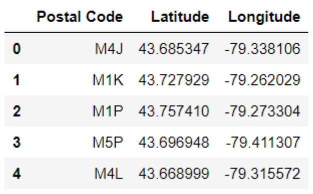


```
geolocator = Nominatim()
latlon = []
for address in addr_series:
    location = geolocator.geocode(address + ", Toronto")
    latlon.append([location.latitude,location.longitude])
lat_lon_df = pd.DataFrame(data = latlon)
lat_lon_df.columns = ["Latitude", "Longitude"]
```

- **Foursquare** – Nearby Venues Cataloged by Postal Codes
  - Postal Code Coordinates

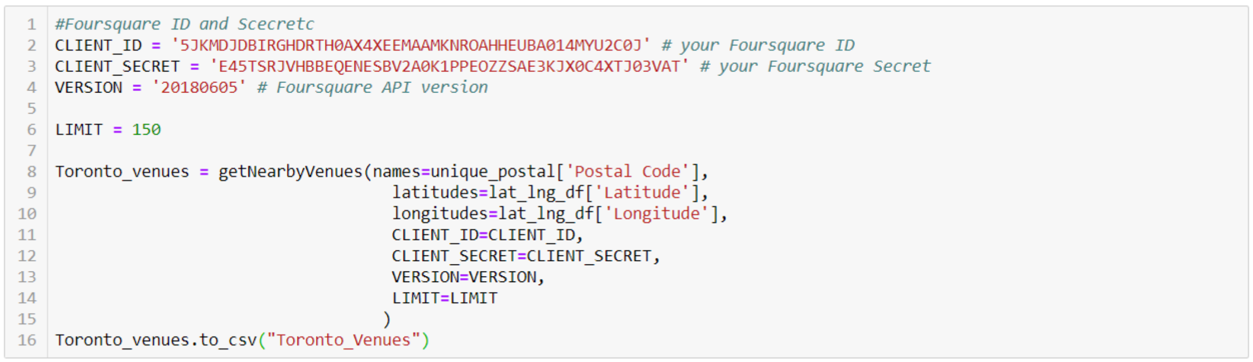


```
lag_lng = pd.Series(map(lambda x: lat_lng_coords[['Latitude', 'Longitude']][lat_lng_coords['Postal Code'] == x].values.tolis
lag_lng_df = pd.DataFrame(columns=['Latitude', 'Longitude'], data = list(lag_lng))
lag_lng_df = pd.concat([unique_postal, lag_lng_df], axis=1, sort=False)[['Postal Code', 'Latitude', 'Longitude']]
lag_lng_df.head()
```

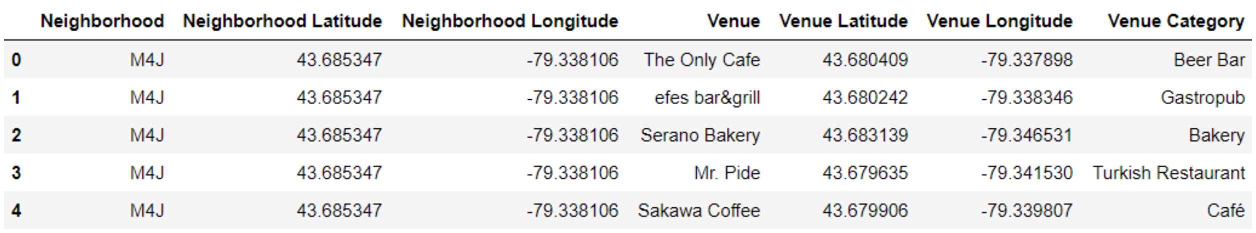


	Postal Code	Latitude	Longitude
0	M4J	43.685347	-79.338106
1	M1K	43.727929	-79.262029
2	M1P	43.757410	-79.273304
3	M5P	43.696948	-79.411307
4	M4L	43.668999	-79.315572

  - Venues Dataframe



```
#Foursquare ID and Scecretc
CLIENT_ID = '5JKMDJDBIRGHDRTH0AX4XEEMAAMKNROAHHEUBA014MYU2C0J' # your Foursquare ID
CLIENT_SECRET = 'E45TSRJVHBBEQENESBV2A0K1PPEOZZSAE3KJX0C4XTJ03VAT' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version

LIMIT = 150

Toronto_venues = getNearbyVenues(names=unique_postal['Postal Code'],
                                   latitudes=lat_lng_df['Latitude'],
                                   longitudes=lat_lng_df['Longitude'],
                                   CLIENT_ID=CLIENT_ID,
                                   CLIENT_SECRET=CLIENT_SECRET,
                                   VERSION=VERSION,
                                   LIMIT=LIMIT
                                  )
Toronto_venues.to_csv("Toronto_Venues")
```



	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	M4J	43.685347	-79.338106	The Only Cafe	43.680409	-79.337898	Beer Bar
1	M4J	43.685347	-79.338106	efes bar&grill	43.680242	-79.338346	Gastropub
2	M4J	43.685347	-79.338106	Serano Bakery	43.683139	-79.346531	Bakery
3	M4J	43.685347	-79.338106	Mr. Pide	43.679635	-79.341530	Turkish Restaurant
4	M4J	43.685347	-79.338106	Sakawa Coffee	43.679906	-79.339807	Café