Capstone Project - (week 5)

Applied Data Science Capstone by IBM/Coursera

Table of contents

Introduction: Business Problem

In this project we will try to find an optimal location to open a new pet store, pet service, or pet cafe in Singapore As of year 2018, the median Singaporean household income is 9,293 which provides spending capacity on domestic items.

Looking around the world, Singapore has it unique culture favor of "East meets West". This country established less than a century ago in 1965. Singapore has several distinct ethnic neighborhoods, including Katong, Kampong Glam, Geylang Serai, Chinatown and Little India[1]. As you can imagine such different ethic combination, it provides a breeding ground of culture melting pot. Before Singapore independent, British ruled Singapore between 1945 to 1964. A western culture introduced by that period of ruling.

Data

Based on definition of our problem, factors that will influence our decision are:

- number of existing pet store, pet service, or pet cafe in the neighborhood.
- number of supermarket which sell pet related product such as pet food.
- Potential new pet store distance between to existing pet store, pet service, or pet cafe in the neighborhood, if any

Obtaining initial data methodology.

- Initial geoloactionswere determinant from postal codes which can be found at file SG.zip geonames.org (http://download.geonames.org).
- Guide To Online Pet Store & Pet Shop Directory In Singapore (https://www.clubpets.com.sg/distribution-pet-shop/). This website provide a list of current pet store in Singapore.
- Singapore government data set List of Licensed Vet Centres (https://data.gov.sg/dataset/list-of-licensed-vet-centres). (File name: list-of-licensed-vet-centres.zip)

Following data sources will be needed to extract/generate the required information:

- More than one neighborhood can exist in several postal codes. For example, in the dataframe above, you will notice that neighborhood Straits Boulevard is listed twice and has two 'Latitude' and 'Longitude': Straits Boulevard appeared twice in geolocation. These two rows will be combined into one row with the 'Latitude', 'Longitude' mean value. The postal code will be consolidated and dropped here which it doesn't provide necessary means for further analysis.
- Foursquare developer account provides 500 Premium call (https://developer.foursquare.com/docs/api/troubleshooting/rate-limits) per day. As this limit, an further merge or cut down numbers of Neighborhood by Latitude, Longitude between two points within 1km is necessary.
- Number of pet store obtained from Foursquare developer account comparing to the local data source such as "Guide To Online Pet Store & Pet Shop Directory In Singapore" was less. It might be the limitation from the type of developer account. We are not dived into the different here.

Neighborhood Candidates

- We will calculate three different proposal against to area supermarket which sell pet related product and existing pet store distance. (1). Base on Foursquare developer account data source. (2) Base on "Guide To Online Pet Store & Pet Shop Directory In Singapore" data source. (3) Base on consolidated postal codes area. The new location proposal rule is find a list shortest pet related product and existing pet store distance "d" then select the maximum value in list "d" [max(list(d)].

Methodology

1. Download and Explore Dataset

Neighborhood has a total 306 neighborhoods after consolidated from 12,115 postal codes. It contains latitude and longitude coordinates of each neighborhood.

For your convenience, I downloaded the files and placed it on the server, so you can simply run a `wget` command and access the data. So let's go ahead and do that.

In [2]:

```python
import folium # map rendering library

import numpy as np # library to handle data in a vectorized manner

import json # library to handle JSON files

import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas dataframe

# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors

# import k-means from clustering stage
from sklearn.cluster import KMeans

import pickle # libary to handle files (JSON or pickle)

print('Libraries imported.')
```

```
Libraries imported.
```

Singpore Postal code dataset

Download postal code from geonames.org

In [ ]:

```
from zipfile import ZipFile

print('Beginning file download with wget module')

!wget -q -O 'geolocation/Singpostal.zip' http://download.geon
ames.org/export/zip/SG.zip  #Use Unix command Wget

print("Waiting.... unzip file")
with ZipFile('geolocation/Singpostal.zip', 'r') as zipObj: #
Create a ZipFile Object and load sample.zip in it
   zipObj.extractall('geolocation/')  # Extract all the conte
nts of zip file in different directory
print('done')
```

Load and explore the data

Next, let's load the data.

In [5]:

```
# loading data
import pandas as pd # library for data analsysis
pd.set_option('display.max_columns', 20)
pd.set_option('display.max_rows', None)

path='geolocation/SG.txt'
df = pd.read_csv(path,sep='\t', header=None)
```

In [6]:

```
# drop the NaN data columns
df.drop([0,3,4,5,6,7,8,11], axis=1, inplace=True)
#change cloumns name. Since it was not exist before, here jus
t pass the cloumn
df.columns = ['postalcode', 'Neighborhood', 'Latitude', 'Long
itude']
df.head()
```

Out[6]:

	postalcode	Neighborhood	Latitude	Longitude
0	18906	Straits Boulevard	1.2758	103.8496
1	18907	Straits Boulevard	1.2749	103.8517
2	18910	Marina Gardens Drive	1.2796	103.8690
3	18915	Central Boulevard	1.2737	103.8601
4	18916	Central Boulevard	1.2798	103.8515

Download a Singapore pet stores list from clubpets.com.sg

To obtain pet store table from clubpets.com.sg, we will do a web page scraping using Pandas read_html and Requests API.
Let's take a quick look at the data.

In [9]:

```
# Code inspríted by
# https://stackoverflow.com/questions/10556048/how-to-extract
-tables-from-websites-in-python/44506462
# scape Singpore pet shop data from website: https://www.club
pets.com.sg/distribution-pet-shop/

import requests
import pandas as pd

url = 'https://www.clubpets.com.sg/distribution-pet-shop/'
df_pet= pd.read_html(requests.get(url).content)[-1]
df_pet.drop(columns='#',inplace=True)
df_pet.head()
#df_pet.to_csv('SingporePetshop.csv')
```

Out[9]:

	Company name	Address	Contact no
0	All About Pets	27 Sembawang Rd S(779080)	64531160
1	All Pets	219 Jln Kayu #01-01 S(799442)	64813700
2	Alphapets (East)	158 Bedok South Ave 3 #01-591 S(460084)	62449868
3	Animal Supermart	Blk 722 Clementi West Street 2 #01-166 S(120722)	96405823
4	Aquapet Centre (Bukit Timah)	1 Jln Anak Bukit #B2-07 Bukit Timah Plaza S(58...	64667175

In [16]:

```
df_pet.columns
```

Out[16]:

```
Index(['Company name', 'Address', 'Contact no', 'postalcode'],
dtype='object')
```

In [17]:

```
# preparae address in same format Then extract postal code an
d save it in df_p dataframe
df_pet['Address'] = df_pet['Address'].str.replace('S\(','Sing
apore(')\
.str.replace('Singapore ','Singapore(').str.replace('off Suns
et Way\)','')

df_pet['postalcode'] = df_pet.Address.str[-8:]
df_pet['postalcode'] = df_pet['postalcode'].str.replace
('\(',' ',regex=True).replace('\)','',regex=True ).replace('e
',' ',regex=True )
```

In [18]:

```
#check dataframe types before to do inner join
#print(df_pet.dtypes,df.dtypes)

#since both postalcode in diferent types, a type converting w
ill preform from object to int64
df_pet['postalcode'] =df_pet.postalcode.astype(int)
```

In [19]:

```
# some of postal area has more than one pet shop.  Use groupb
y() 'postalcode' and count() pet store.
# create a new dataframe for the pet store count in each pots
al area
df_petStoreCount=df_pet.groupby('postalcode')['Address'].coun
t().reset_index(name="pet store count")
df_petStoreCount.shape
```

Out[19]:

```
(114, 2)
```

Here, we perpare geolocation and existing pet storelist

In [20]:

```
# obtain geolocation position from df
# merge df, df_pet, & df_petStoreCount
df_m = pd.merge(left=df_pet,right=df, left_on='postalcode', r
ight_on='postalcode')
df_m = pd.merge(left=df_m,right=df_petStoreCount, left_on='po
stalcode', right_on='postalcode')

# add type column and mark '0' as it is pet store location da
taset
df_m['Type']= 0
df_m.shape
```

Out[20]:

```
(118, 9)
```

```
In [21]: df_m.head()
```

```
Out[21]:
```

	Company name	Address	Contact no	postalcode	Neighborhood	Latitude	Long
0	All About Pets	27 Sembawang Rd Singapore(779080)	64531160	779080	Sembawang Road	1.4031	103.
1	All Pets	219 Jln Kayu #01-01 Singapore(799442)	64813700	799442	Jalan Kayu	1.3956	103.
2	Alphapets (East)	158 Bedok South Ave 3 #01-591 Singapore(460084)	62449868	460084	Bedok North Street 4	1.3318	103.
3	K1 Pet Shop	84 Bedok North St 4 #01-09 Singapore(460084)	64478588	460084	Bedok North Street 4	1.3318	103.
4	Animal Supermart	Blk 722 Clementi West Street 2 #01-166 Singapo...	96405823	120722	Clementi West Street 2	1.3029	103.

Let work on consolidate postal codes.

Let examine Singapore potsal code. If one examining postal code, one would find some strees divides into (total 121,154 postal code) more than one postal code. It give an opportunity to reduce some of postal code for this project.

```
In [22]: df.shape
```

```
Out[22]: (121154, 4)
```

In [23]:
```
# use groupby on Neighborhood and average 'Latitude','Longitu
de' value to reduce number of 'Latitude' and 'Longitude' valu
e
df_pos_g = df.groupby('Neighborhood')['Latitude','Longitude
'].mean().reset_index()
print('{0}\n\nDataframe size= {1}'.format(df_pos_g.head(),df_
pos_g.shape))
```

```
    Neighborhood  Latitude   Longitude
0  Abingdon Road  1.368025  103.980287
1     Adam Drive  1.336363  103.814787
2      Adam Park  1.331032  103.812989
3      Adam Road  1.326481  103.813350
4      Adis Road  1.301325  103.847675

Dataframe size= (3863, 3)
```

Foursquare developer account provides [500 Premium call (https://developer.foursquare.com/docs/api/troubleshooting/rate-limits)](https://developer.foursquare.com/docs/api/troubleshooting/rate-limits) per day. As this limit, to merge or cut down numbers of Neighborhood by Latitude, Longitude between two points within 1km is necessary.

Sorting Latiude and longittude here to combin Neighorhoods by neighborhood distance

In [24]:

```python
# helper function
# the diameter of the Earth is 12,742 km, unit result in this calculation is Km
# refernce: http://mathforum.org/library/drmath/view/51816.html for latitude, longitude equation &
# https://stackoverflow.com/questions/42686300/how-to-check-if-coordinate-inside-certain-area-python

from math import cos, asin, sqrt
def distance(lat_set1, lon_set1, lat_set2, lon_set2):
    for lat1, lon1, lat2, lon2 in zip(lat_set1, lon_set1, lat_set2, lon_set2):
        p = 0.017453292519943295     #Pi/180
        a = 0.5 - cos((lat2 - lat1) * p)/2 + cos(lat1 * p) * cos(lat2 * p) * (1 - cos((lon2 - lon1) * p)) / 2
    return 12742 * asin(sqrt(a)) #distance_list #2*R*asin...; united in Km

# refernce: https://stackoverflow.com/questions/34562261/get-pairwise-iterator-with-additional-item-in-the-end
from itertools import tee
def pairwise(iterable):
    "s -> (s0,s1), (s1,s2), (s2, s3), ..."
    a, b = tee(iterable)
    next(b, None)
    return zip(a, b)
```

A better way to combin the Neighborhood is create a circle of insterst area then test near point between the center point. Code to check it is below from [https://stackoverflow.com/questions/42686300/how-to-check-if-coordinate-inside-certain-area-python (https://stackoverflow.com/questions/42686300/how-to-check-if-coordinate-inside-certain-area-python)](https://stackoverflow.com/questions/42686300/how-to-check-if-coordinate-inside-certain-area-python)

In [25]:

```
numRun=centerControl=0;
borough=1.0;  #0.5 Km borough range
start = True;

print("Wating... Calculating near Neighborhood and consoildate within {0} Km range".format(borough))
while start == True:
    rowdroplist=[]
    for (i1, row1), (i2, row2) in pairwise(df_pos_g[numRun:].iterrows()): # given dataframe
        if centerControl==0: center=row1;  # set center point on row1 on each iterate
        else: row1=center;
        centerControl+=1;
        NeighborhoodDis = distance(lat_set1=[row1['Latitude']], lon_set1= [row1['Longitude']],\
                                   lat_set2=[row2['Latitude']], lon_set2= [row2['Longitude']])

        if NeighborhoodDis<= borough:
            rowdroplist.append(i2)
            combin_Neighborhood=', '.join((row1['Neighborhood'],row2['Neighborhood']))  # combin Neighborhood
            df_pos_g.at[i1,'Neighborhood']=combin_Neighborhood  # change Neighborhood description

    numRun += 1;
    df_pos_g.drop(rowdroplist, axis=0, inplace=True)
    centerControl=0; # reset centerControl to 0;
    if numRun > df_pos_g.shape[0]: start = False;  # control while loop
    df_pos_g.reset_index(drop=True, inplace=True)
    if numRun %50==1: print("Neighborhood consoildated {1} at counter {0}, Neighborhood remaind:{2}" \
    .format(numRun,len(rowdroplist),df_pos_g.shape[0]))
print("Number of Neighborhood after consolidating={0}'\n'".format(df_pos_g.shape))
```

```
Wating... Calculating near Neighborhood and consoildate within
1.0 Km range
Neighborhood consoildated 9 at counter 1, Neighborhood remain
d:3854
Neighborhood consoildated 11 at counter 51, Neighborhood remai
nd:2296
Neighborhood consoildated 8 at counter 101, Neighborhood remai
nd:1545
Neighborhood consoildated 11 at counter 151, Neighborhood rema
ind:919
Neighborhood consoildated 8 at counter 201, Neighborhood remai
nd:594
Neighborhood consoildated 0 at counter 251, Neighborhood remai
nd:458
Neighborhood consoildated 10 at counter 301, Neighborhood rema
ind:323
Number of Neighborhood after consolidating=(306, 3)'
'
```

Finally, postal code reduced to size of 306 in the neighborhood.
A marker will be added into dataframe for visualization.

In [26]:
```
# add type column to mark it as postal code
df_pos_g['Type'] = 1
```

In [27]:
```
df_pos_g.head()
```

Out[27]:

	Neighborhood	Latitude	Longitude	Type
0	Abingdon Road	1.368025	103.980287	1
1	Adam Drive, Adam Park	1.336363	103.814787	1
2	Adam Road	1.326481	103.813350	1
3	Adis Road	1.301325	103.847675	1
4	Admiralty Drive, Admiralty Lane	1.449964	103.816195	1

Visualize data

A visualization on postal code and pet store (data from clubpets.com.sg) location on map.

In [28]:

```
# union df_pos_g (postal code) and df_mpet (pet store)
# drop df_m cloumns but keep Neighborhood,Latitude,Longitude,
and Type into new dataframe df_mpet
df_mpet=df_m.drop(['Company name','Address', 'Contact no', 'p
ostalcode','pet store count'],axis=1,)
```

In [29]:

```
df_mpet.head()
```

Out[29]:

	Neighborhood	Latitude	Longitude	Type
0	Sembawang Road	1.4031	103.8176	0
1	Jalan Kayu	1.3956	103.8728	0
2	Bedok North Street 4	1.3318	103.9392	0
3	Bedok North Street 4	1.3318	103.9392	0
4	Clementi West Street 2	1.3029	103.7637	0

In [30]:

```
df_pos_mpet = df_pos_g.append(df_mpet, ignore_index=True)
```

In [31]:

```
df_pos_mpet.shape
```

Out[31]:

```
(424, 4)
```

Lookup Singapore geolocation

In [32]:

```
latitude =df_pos_mpet['Latitude'][0]
longitude = df_pos_mpet['Longitude'][0]
print('The geographical coordinate of Singapore are {}, {}.'.f
ormat(latitude, longitude))
```

```
The geograpical coordinate of Singapore are 1.368025, 103.9802
8749999999.
```

Create Singapore area postal and pet store MAP. Circle spots in 'blue' are the postal code; in 'red' are pet store geo-location. This is not the clustering map!

In [33]:

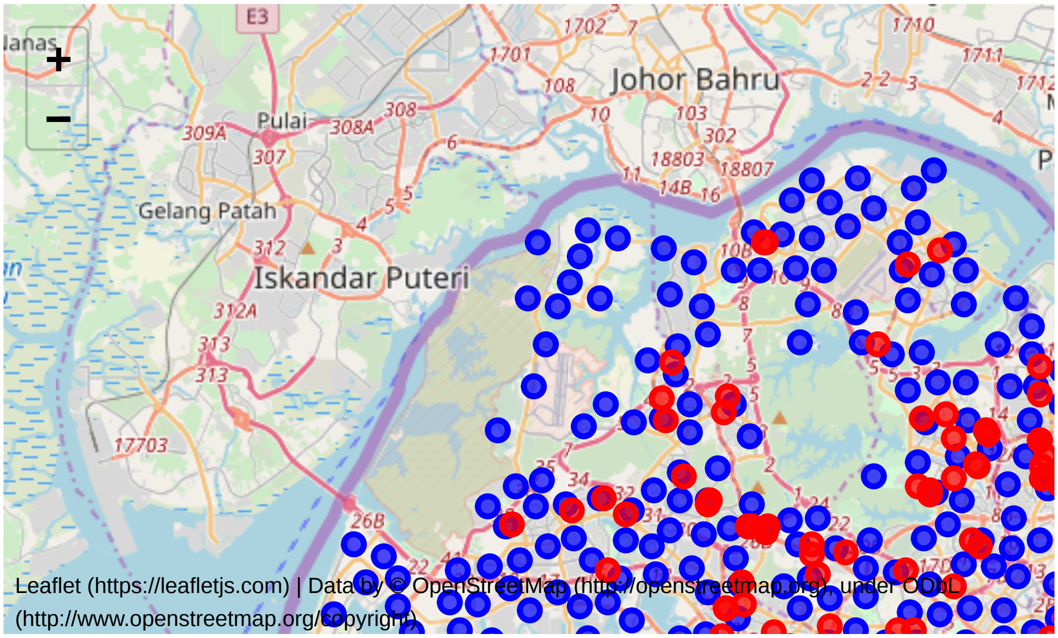
```
# kType=2
# create map of Manhattan using latitude and longitude values
map_SingaporeRegion = folium.Map(location=[latitude, longitude], zoom_start=11)

# set color scheme for the clusters
# x = np.arange(kType)
# ys = [i + x + (i*x)**2 for i in range(kType)]
# colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
# rainbow = [colors.rgb2hex(i) for i in colors_array]
rainbow =['red','blue']

# add markers to map
for lat, lng, label, Type in zip(df_pos_mpet['Latitude'], df_pos_mpet['Longitude'], df_pos_mpet['Neighborhood']\
                                ,df_pos_mpet['Type']):
    #label = folium.Popup(label, parse_html=True)
    label = folium.Popup(str(label) + ' Type ' + str(Type), parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color=rainbow[Type],
        fill=True,
        fill_color=rainbow[Type],
        fill_opacity=0.7).add_to(map_SingaporeRegion)

map_SingaporeRegion
```

Out[33]:



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

Analysis

Foursquare

Now that we have our potsal location candidates, let's use Foursquare API to get info on pet store in each neighborhood.

We're interested in venues in 'pet' category. Forsquare catagoies code on Pet Café:56aa371be4b08b9a8d573508, Pet Service: 5032897c91d4c4b30a586d69, Pet Store:4bf58dd8d48988d100951735. The related venue Categories are Pharmacy:4bf58dd8d48988d10f951735, Animal Shelter:4e52d2d203646f7c19daa8ae, Fishing Store:52f2ab2ebcbc57f1066b8b16.

Foursquare credentials are defined in hidden cell bellow.

In [75]:

```
# yahoo Keys (key was reset when this submit to Github)
CLIENT_ID = 'GExxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxQZ' # your Foursquare ID
CLIENT_SECRET = 'LXxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxHC' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: GExxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxQ
Z
CLIENT_SECRET:LXxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xHC
```

In [76]:

```
# type your answer here
LIMIT = 100 # limit of number of venues returned by Foursquar
e API
radius = 1000 # define radius

# create URL
url = 'https://api.foursquare.com/v2/venues/explore?&client_i
d={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.forma
t(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    latitude,
    longitude,
    radius,
    LIMIT)
url # display URL
```

Out[76]:

```
'https://api.foursquare.com/v2/venues/explore?&client_id=GExxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxQZ&client_secret=
LXxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxHC&v=20180605
&ll=1.368025,103.98028749999999&radius=1000&limit=100'
```

One dataset result from Foursquare for understand it structure. You can see a lot inform in one geolocation point for 1km radius.

In [37]:

```
results = requests.get(url).json()
results
```

Out[37]:

```
{'meta': {'code': 200, 'requestId': '5e2c7486760a7f001b7a3f24
'},
 'response': {'headerLocation': 'Changi',
  'headerFullLocation': 'Changi, Singapore',
  'headerLocationGranularity': 'neighborhood',
  'totalResults': 9,
  'suggestedBounds': {'ne': {'lat': 1.3770250090000091,
    'lng': 103.98927326617574},
   'sw': {'lat': 1.359024990999991, 'lng': 103.9713017338242
3}},
  'groups': [{'type': 'Recommended Places',
    'name': 'recommended',
    'items': [{'reasons': {'count': 0,
       'items': [{'summary': 'This spot is popular',
         'type': 'general',
         'reasonName': 'globalInteractionReason'}]},
      'venue': {'id': '527f2b4611d2d329ca88630e',
       'name': '1942 Alfresco@Changi',
       'location': {'address': '30 Cosford Rd.',
        'lat': 1.3652098257663445,
        'lng': 103.98142729316815,
        'labeledLatLngs': [{'label': 'display',
          'lat': 1.3652098257663445,
          'lng': 103.98142729316815}],
        'distance': 338,
        'postalCode': '499550',
        'cc': 'SG',
        'country': 'Singapore',
        'formattedAddress': ['30 Cosford Rd.', '499550', 'Sing
apore']},
       'categories': [{'id': '4bf58dd8d48988d1c4941735',
         'name': 'Restaurant',
         'pluralName': 'Restaurants',
         'shortName': 'Restaurant',
         'icon': {'prefix': 'https://ss3.4sqi.net/img/categori
es_v2/food/default_',
          'suffix': '.png'},
         'primary': True}],
       'photos': {'count': 0, 'groups': []}},
      'referralId': 'e-0-527f2b4611d2d329ca88630e-0'},
     {'reasons': {'count': 0,
       'items': [{'summary': 'This spot is popular',
         'type': 'general',
         'reasonName': 'globalInteractionReason'}]},
      'venue': {'id': '56e2b449498e951533ab584d',
       'name': '555 Villa Thai',
       'location': {'address': '30 Cosford Road',
        'lat': 1.3652269943159228,
        'lng': 103.98123657619259,
        'labeledLatLngs': [{'label': 'display',
          'lat': 1.3652269943159228,
          'lng': 103.98123657619259}],
```

After review the returned result, the data of interest starting from "['response']['groups'][0]['items']"

Get a helping function "is_pet_shop" which return three catagories: other_catgories, pet_rel, pet_specific

In [38]:

```
# Category IDs corresponding to pet store were taken from Foursquare web site (https://developer.foursquare.com/docs/resources/categories):
pet_categories = ['56aa371be4b08b9a8d573508', '5032897c91d4c4b30a586d69', '4bf58dd8d48988d100951735', '52f2ab2ebcbc57f1066b8b16']
# {Pet Café:56aa371be4b08b9a8d573508, Pet Service:5032897c91d4c4b30a586d69,Pet Store:4bf58dd8d48988d100951735,
# Pharmacy:4bf58dd8d48988d10f951735, Animal Shelter:4e52d2d203646f7c19daa8ae, Fishing Store:52f2ab2ebcbc57f1066b8b16}

def is_pet_shop(categories_name, categories_id, specific_filter=None):
    other_catgories= pet_rel = pet_specific = False
    control_i=0;
    pet_words = ['pet','animal', 'pharmacy', 'shelter','pet beauty salon','fishing store']
     #pet_related=['pharmacy', 'shelter','pet beauty salon']

    category_name = categories_name.lower()
    for w in pet_words:
        if (w in category_name):
            pet_rel =True
            control_i +=1;
            if (categories_id in specific_filter):
                pet_specific = True
                pet_rel = False
    if (control_i==0):
        other_catgories=True
    #print(control_i,other_catgories, pet_rel, pet_specific)
    return other_catgories, pet_rel, pet_specific
```

2. Explore Neighborhoods in Singapore

Let's create a function to repeat the same process to all the neighborhoods in Singapore

Now write code to run the above function on each neighborhood and create new dataframes called regular_venues, pet_related_store,pet_store

In [39]:

```
def getNearbyVenues(names, latitudes, longitudes, radius=100
0):
    venues_list=[]
    i=0;
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(i,name)
        i +=1

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&
client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit=
{}'.format(
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['group
s'][0]['items']

        # return only relevant information for each nearby ve
nue
        venues_list.append([(
            name,
            lat,
            lng,
            v['venue']['categories'][0]['id'],
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['location']['distance'],
          # v['venue']['location']['city'],  Singapore city e
qual contry
            v['venue']['categories'][0]['name']) for v in res
ults])

    nearby_venues = pd.DataFrame([item for venue_list in venu
es_list for item in venue_list])
    nearby_venues.columns = ['Neighborhood',
                             'Neighborhood Latitude',
                             'Neighborhood Longitude',
                             'Venue_category_id',
                             'Venue_name',
                             'Venue Latitude',
                             'Venue Longitude',
                             'Venue distance',
                           # 'Venue City'
```

In [40]:

```
# Try to load from local file system in case we did this befo
re
regular_venues =[]; pet_related_store=[]; pet_store = [];
loaded = False
try:
    with open('regular_venues.pkl', 'rb') as f:
        regular_venues = pickle.load(f)
    with open('pet_related_store.pkl', 'rb') as f:
        pet_related_store=pickle.load(f)
    with open('pet_store.pkl', 'rb') as f:
        pet_store = pickle.load(f)
    print('Pet store data loaded.')
    loaded = True
except:
    pass


# If load failed use the Foursquare API to get the data
if not loaded:
    venues= getNearbyVenues(names=df_m['Neighborhood'],
                            latitudes=df_m['Latitude'],
                            longitudes=df_m['Longitude']
                           )

    #print(venues)
    for venue in venues.iterrows(): # create a tuple
        venue=venue[1] # cloumn 1 is a of the venue, afterwar
e venue become a list have 8 element

        venue_categories_id = venue[3]
        venue_name = venue[4]
        venue_categories = venue[8]
        venue_lat = venue[2]
        venue_lon= venue[3]
        venue_address = venue[0]
        venue_distance = venue[7]
        are_other_catgories, pet_related_shop, are_pet_specif
ic_shops = is_pet_shop(venue_categories, venue_categories_i
d,\

specific_filter=pet_categories) #venue_id

        if are_other_catgories==True:
            regular_venues.append(venue)
        if pet_related_shop==True:
            pet_related_store.append(venue) #[venue_categorie
s_id]=venue
        if are_pet_specific_shops==True:
            pet_store.append(venue)  #[venue_categories_id]=v
enue
        del (are_other_catgories, pet_related_shop, are_pet_s
Pet store data loaded.
```

In [41]:

```
# remove duplicated rows
regular_venues= pd.DataFrame(regular_venues)
pet_related_store= pd.DataFrame(pet_related_store)
pet_store = pd.DataFrame(pet_store)

rv_indexNames = regular_venues[regular_venues.duplicated(keep
='first')].index
pr_indexNames= pet_related_store[pet_related_store.duplicated
(keep='first')].index
ps_indexNames= pet_store[pet_store.duplicated(keep='first')].
index

regular_venues.drop(rv_indexNames , inplace=True)
pet_related_store.drop(pr_indexNames , inplace=True)
pet_store.drop(ps_indexNames , inplace=True)

pet_store_g=pet_store.groupby('Neighborhood')['Venue Category
'].count()
pet_store_g=pd.DataFrame(pet_store_g)  # series object conver
t to DataFrame object
pet_store_g.columns=['# of store count']
```

In [42]:

```
print('Total number of regular venus:', len(regular_venues))
print('Total number of pet_related store:', len(pet_related_s
tore))
print('Total number of pet_store:', len(pet_store))
print('Percentage of pet_store: {:.2f}%'.format(len(pet_stor
e) / len(regular_venues) * 100))
print('Average number of pet_store in neighborhood:', pet_sto
re_g['# of store count'].mean())
```

```
Total number of regular venus: 7610
Total number of pet_related store: 20
Total number of pet_store: 37
Percentage of pet_store: 0.49%
Average number of pet_store in neighborhood: 1.423076923076923
1
```

In [43]:
```
pet_store.head()
```

Out[43]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue_category_id	Venue_
73	Jalan Kayu	1.3956	103.8728	4bf58dd8d48988d100951735	Petg
367	Jalan Anak Bukit	1.3387	103.7786	4bf58dd8d48988d100951735	Paws
778	Yio Chu Kang Road	1.3582	103.8749	4bf58dd8d48988d100951735	Tl
832	Clementi Street 11	1.3219	103.7707	4bf58dd8d48988d100951735	P
857	Clementi Street 11	1.3219	103.7707	4bf58dd8d48988d100951735	Cle Fl Aqu (

Let's now go over our neighborhood locations and get nearby pet_store; we'll also maintain a dictionary of all found Venues

In [44]:
```
pet_store_g.head(2)
```

Out[44]:

	# of store count
Neighborhood	
Anamalai Avenue	1
Ang Mo Kio Avenue 1	1

In [45]:
```
pet_store = pd.merge(left=pet_store, right=pet_store_g, left_
on='Neighborhood', right_on='Neighborhood')
```

In [46]:
```
pet_store.head(2)
```

Out[46]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue_category_id	Venue_na
0	Jalan Kayu	1.3956	103.8728	4bf58dd8d48988d100951735	S Petgan
1	Jalan Anak Bukit	1.3387	103.7786	4bf58dd8d48988d100951735	Paws " F

Explore if an area does not have any pet store or pet service but already has a pet_related busniess (e.g. selling dog/cat food busniess).

In [47]:
```
# find area has pet_store and pet_related busniess list
pet_sr_m = pd.merge(left=pet_store['Neighborhood'],\
                    right=pet_related_store[['Neighborhood','
Venue_name','Venue Latitude','Venue Longitude']], \
                    left_on='Neighborhood', right_on='Neighbo
rhood', how='outer', indicator=True)
```

In [48]:
```
pet_sr_m.head(2)
```

Out[48]:

	Neighborhood	Venue_name	Venue Latitude	Venue Longitude	_merge
0	Jalan Kayu	NaN	NaN	NaN	left_only
1	Jalan Anak Bukit	NaN	NaN	NaN	left_only

In [50]:
```
pet_sr_m_ro=pet_sr_m[pet_sr_m['_merge']=='right_only']
pet_sr_m_ro_w=pet_sr_m_ro
```

Now calulate the nearest pet store distance bewteen all Waston list location above then select the farest distance in calculated distance result list for the potient new pet store location.

Visualize potient pet store locations

In [73]:
```
pet_sr_m_ro_w['Type']=2
pet_sr_m_ro_w.drop(columns=['Venue_name','_merge'], inplace=True)
pet_sr_m_ro_w.columns=['Neighborhood', 'Latitude', 'Longitude', 'Type']
```

In [52]:
```
# prepare pet_store dataframe to match with pet_sr_m_ro_w
pet_store_col_d=pet_store.drop(columns=['Venue_category_id', 'Venue_name', 'Venue Latitude', 'Venue Longitude',
                                       'Venue distance', 'Venue Category', '# of store count'])
pet_store_col_d['Type']=1
pet_store_col_d.columns=['Neighborhood', 'Latitude', 'Longitude', 'Type']
pet_store_col_d.head()
```

Out[52]:

	Neighborhood	Latitude	Longitude	Type
0	Jalan Kayu	1.3956	103.8728	1
1	Jalan Anak Bukit	1.3387	103.7786	1
2	Yio Chu Kang Road	1.3582	103.8749	1
3	Yio Chu Kang Road	1.3615	103.8738	1
4	Clementi Street 11	1.3219	103.7707	1

In [53]:
```
# resue df_pos_g['Type'] = 1
df_pos_m_potientPetStore = pet_store_col_d.append(pet_sr_m_ro_w, ignore_index=True)
```

In [54]:

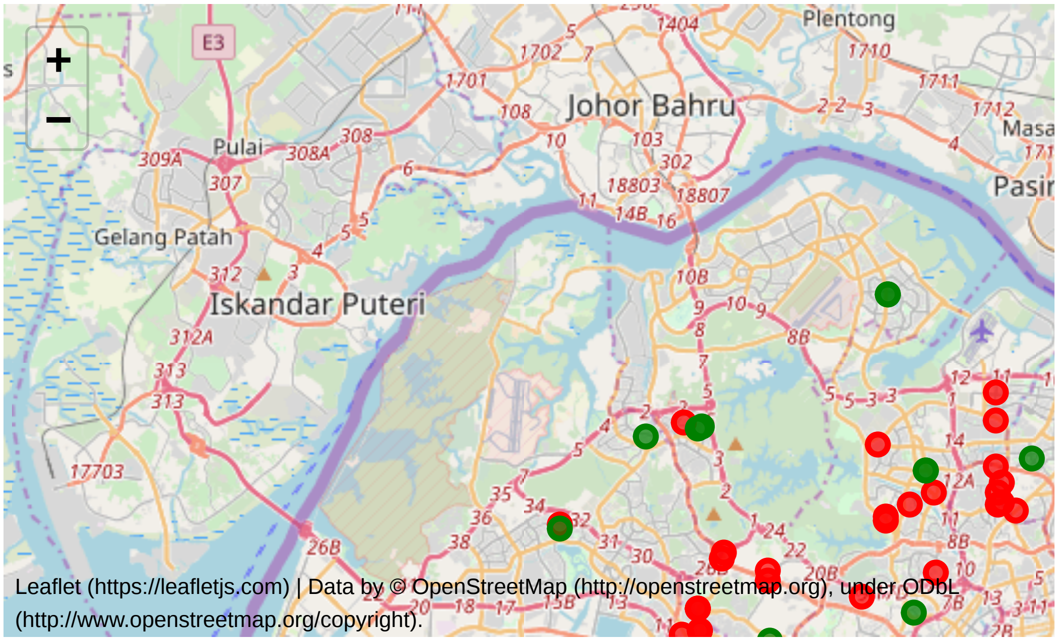
```
map_SingaporeRegion_potiental_location = folium.Map(location=[latitude, longitude], zoom_start=11)

rainbow =['blue','red','green']

# add markers to map
for lat, lng, label, Type in zip(df_pos_m_potientPetStore['Latitude'], df_pos_m_potientPetStore['Longitude'],
                                  df_pos_m_potientPetStore['Neighborhood'],df_pos_m_potientPetStore['Type']):
    #label = folium.Popup(label, parse_html=True)
    label = folium.Popup(str(label) + ' Type ' + str(Type), parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color=rainbow[Type],
        fill=True,
        fill_color=rainbow[Type],
        fill_opacity=0.7).add_to(map_SingaporeRegion_potiental_location)

map_SingaporeRegion_potiental_location
```

Out[54]:


+
−
E3
Plentong
Johor Bahru
Pulai
Masa
Pasir
Gelang Patah
Iskandar Puteri
Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

Results and Discussion

Distance list calculation

Case 1: FourSquare pet stores distance between related pet venue (supermarket)

In [55]:
```
potientialDis=[]
for (poti_Neighborhood, center_lat, center_lon) in zip(pet_sr
_m_ro_w['Neighborhood'],pet_sr_m_ro_w ['Latitude'],pet_sr_m_r
o_w ['Longitude']):
    tempDis=99999; #initial abitrial distance
    for (pet_Neighborhood,ps_lat,ps_lon) in zip(pet_store_col
_d['Neighborhood'],pet_store_col_d['Latitude'],pet_store_col_
d['Longitude']):
        if (center_lat!=ps_lat) and (center_lon!=ps_lon):
            NeighborhoodDis = distance(lat_set1=[center_lat],
lon_set1= [center_lon],
                                       lat_set2=[ps_lat], lon
_set2= [ps_lon])
            if (NeighborhoodDis > 0.0) and (tempDis>Neighborh
oodDis):
                tempDis = NeighborhoodDis
    #print(poti_Neighborhood,pet_Neighborhood, tempDis)
    potientialDis.append([poti_Neighborhood, center_lat, cent
er_lon, tempDis])

finialDF1=pd.DataFrame(potientialDis, columns = ['Neighborhoo
d', 'Latitude', 'Longitude', 'Dis'])
```

Base on Foursquare data, a new pet store proposed location which already has supermarket but pet store no existing within approximate 5.5Km at Yishun Street 22 or Yishun Avenue 5. Yishun Street 22 or Yishun Avenue 5 are close proximate to each other less than 1Km.

In [56]:
```
finialDF1[finialDF1['Dis']==finialDF1['Dis'].max()]
```

Out[56]:

	Neighborhood	Latitude	Longitude	Dis
2	Yishun Street 22	1.429384	103.835667	5.58132
4	Yishun Avenue 5	1.429384	103.835667	5.58132

Case 2: "Guide To Online Pet Store & Pet Shop Directory In Singapore" pet stores distance between related pet venue (supermarket)

In [58]:

```
potientialDis=[]
for (poti_Neighborhood, center_lat, center_lon) in zip(pet_sr
_m_ro_w['Neighborhood'],pet_sr_m_ro_w ['Latitude'],pet_sr_m_r
o_w ['Longitude']):
    tempDis=99999; #initial abitrial distance
    for (pet_Neighborhood,ps_lat,ps_lon) in zip(df_mpet['Neig
hborhood'],df_mpet['Latitude'],df_mpet['Longitude']):
    #for (pet_Neighborhood,ps_lat,ps_lon) in zip(pet_store_co
l_d['Neighborhood'],pet_store_col_d['Latitude'],pet_store_col
_d['Longitude']):
        if (center_lat!=ps_lat) and (center_lon!=ps_lon):
            NeighborhoodDis = distance(lat_set1=[center_lat],
lon_set1= [center_lon],
                                       lat_set2=[ps_lat], lon
_set2= [ps_lon])
            if (NeighborhoodDis > 0.0) and (tempDis>Neighborh
oodDis):
                tempDis = NeighborhoodDis
    #print(poti_Neighborhood,pet_Neighborhood, tempDis)
    potientialDis.append([poti_Neighborhood, center_lat, cent
er_lon, tempDis])

finialDF2=pd.DataFrame(potientialDis, columns = ['Neighborhoo
d', 'Latitude', 'Longitude', 'Dis'])
```

Base on case 2 data, a new pet store proposed location which already has supermarket but pet store no existing within approximate 0.98Km at Choa Chu Kang Avenue 1.

In [59]:

```
finialDF2[finialDF2['Dis']==finialDF2['Dis'].max()]
```

Out[59]:

	Neighborhood	Latitude	Longitude	Dis
10	Choa Chu Kang Avenue 1	1.380632	103.752645	0.988623

Case 3: Pet stores distance between consolidate postal codes which explore the non pet store area.

In [61]:

```
# use postal code to determine the best new pet store location regardless other
# df_pos_g
potientialDis=[]
for (poti_Neighborhood, center_lat, center_lon) in zip(df_pos_g['Neighborhood'],df_pos_g ['Latitude'],df_pos_g ['Longitude']):
    tempDis=99999; #initial abitrial distance
    for (pet_Neighborhood,ps_lat,ps_lon) in zip(df_mpet['Neighborhood'],df_mpet['Latitude'],df_mpet['Longitude']):
    #for (pet_Neighborhood,ps_lat,ps_lon) in zip(pet_store_col_d['Neighborhood'],pet_store_col_d['Latitude'],pet_store_col_d['Longitude']):
        if (center_lat!=ps_lat) and (center_lon!=ps_lon):
            NeighborhoodDis = distance(lat_set1=[center_lat], lon_set1= [center_lon],
                                       lat_set2=[ps_lat], lon_set2= [ps_lon])
            if (NeighborhoodDis > 0.0) and (tempDis>NeighborhoodDis):
                tempDis = NeighborhoodDis
    #print(poti_Neighborhood,pet_Neighborhood, tempDis)
    potientialDis.append([poti_Neighborhood, center_lat, center_lon, tempDis])

finialDF3=pd.DataFrame(potientialDis, columns = ['Neighborhood', 'Latitude', 'Longitude', 'Dis'])
```

Base on case 3 data, a new pet store proposed location farthest distance that non-of pet store in the area is approximate 12.55Km away from any existing established pet store at Pulau Tekong Besar.

In [63]:

```
#finialDF.cloumns=['Neighborhood', 'Latitude', 'Longitude', 'Dis']
finialDF3[finialDF3['Dis']==finialDF3['Dis'].max()]
```

Out[63]:

	Neighborhood	Latitude	Longitude	Dis
254	Pulau Tekong Besar	1.4014	104.059	12.555692

Dissussion:

Reviewing all three cases, the proposal new pet store location was very different. Although the methodology is the same, the initial data source would make a big impact on the result. In this comprehensive analysis, we have to caution getting data source from commercial data provide which limited a quality of data at least on this study. Before getting a set of data from commercial site, one should understand the site restriction on different kinds of account limitation.
We can scrape case 1 result. We know the data set pet store is much smaller. We will keep case 2 and case 3 for further study to including more factors into consideration such as populate, social acceptance on keeping pet, income level, government veteran offices. It is interesting to see more factors adding into new site selecting process might affect the finial result.
Next step is to add government veteran office data which can obtain at [List of Licensed Vet Centres (https://data.gov.sg/dataset/list-of-licensed-vet-centres)](https://data.gov.sg/dataset/list-of-licensed-vet-centres) on the analysis.

Visualize result

In [66]:
```
finialPotientialPetStoreLocation=finialDF1[finialDF1['Dis']==
finialDF1['Dis'].max()]
finialPotientialPetStoreLocation=finialPotientialPetStoreLoca
tion.append(finialDF2[finialDF2['Dis']==finialDF2['Dis'].max
()],\

ignore_index=True)
finialPotientialPetStoreLocation=finialPotientialPetStoreLoca
tion.append(finialDF3[finialDF3['Dis']==finialDF3['Dis'].max
()],\

ignore_index=True)
#df_pos_mpet = df_pos_g.append(df_mpet, ignore_index=True)
```

In [68]:

```
finialPotientialPetStoreLocation['Type']=2
finialPotientialPetStoreLocation = finialPotientialPetStoreLo
cation.append(df_mpet, ignore_index=True)
finialPotientialPetStoreLocation.head()
```

Out[68]:

	Dis	Latitude	Longitude	Neighborhood	Type
0	5.581320	1.429384	103.835667	Yishun Street 22	2
1	5.581320	1.429384	103.835667	Yishun Avenue 5	2
2	0.988623	1.380632	103.752645	Choa Chu Kang Avenue 1	2
3	12.555692	1.401400	104.059000	Pulau Tekong Besar	2
4	NaN	1.403100	103.817600	Sembawang Road	2

'red' marks are existing pet store
'green' marks are proposed new pet store location according analysis from three cases

In [69]:

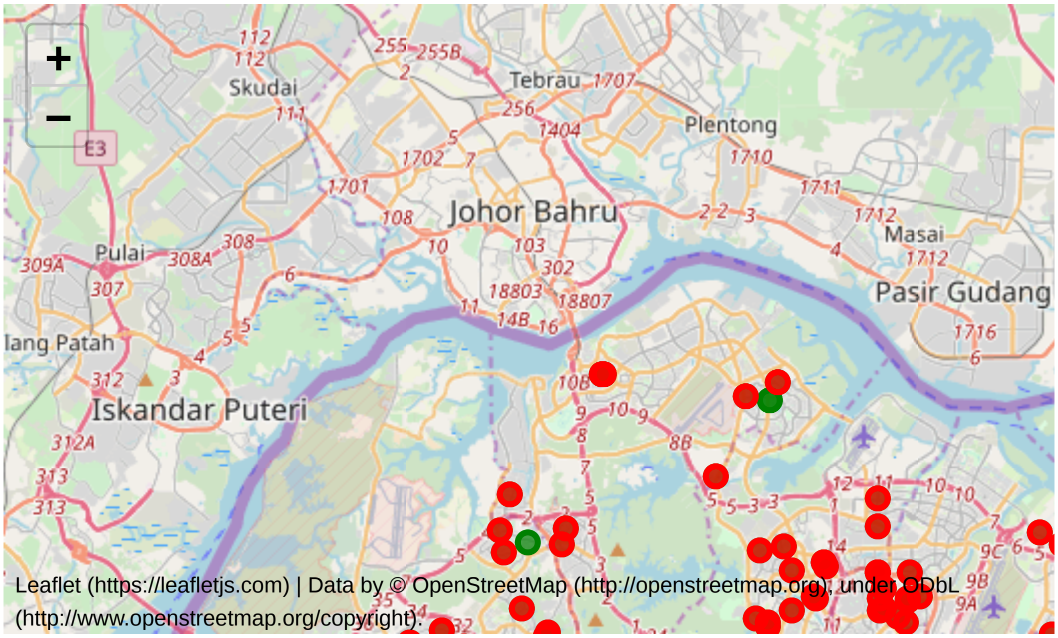
```
map_SingaporeRegion_potiental_finial_location = folium.Map(lo
cation=[latitude, longitude], zoom_start=11)

rainbow =['red','blue','green']

# add markers to map
for lat, lng, label, Type in zip(finialPotientialPetStoreLoca
tion['Latitude'], finialPotientialPetStoreLocation['Longitude
'],
                                  finialPotientialPetStoreLoca
tion['Neighborhood'],finialPotientialPetStoreLocation['Type
']):
    #label = folium.Popup(label, parse_html=True)
    label = folium.Popup(str(label) + ' Type ' + str(Type), p
arse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color=rainbow[Type],
        fill=True,
        fill_color=rainbow[Type],
        fill_opacity=0.7).add_to(map_SingaporeRegion_potienta
l_finial_location )

map_SingaporeRegion_potiental_finial_location
```

Out[69]:



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

Conclusion

It is a question of blue ocean or red sea strategy. We know case 1 is not valid result. It scrapes on the study. Case 2 result is in very close approximation of already established pet store location, 0.95km away. It will create a high competition and tension between new store. More data on other factors has to include on the study before a finial decision to make. Case 3 is a easy to accept. Using the consolidated postal code against the existing pet store gives the most satisfied result(blue ocean). However, a question should raise after visualizing between existing pet store location on map. One can easily see that most of established pet stores are very centralizer in the middle ranger. Outer perimeter area (outskirts area) no matter east, west or south, non of pet store is in that area. Case 3 suggests area exactly locating in north east outskirts. This is a cautionary. Other factors should be understood before the finalize selection.

Reference:

1. https://en.wikipedia.org/wiki/Culture_of_Singapore (https://en.wikipedia.org/wiki/Culture_of_Singapore).
2. https://en.wikipedia.org/wiki/Postal_codes_in_Singapore (https://en.wikipedia.org/wiki/Postal_codes_in_Singapore)
3. geolocation http://download.geonames.org/export/zip/ (http://download.geonames.org/export/zip/)
4. Waston supermarket https://www.watsons.com.sg/ (https://www.watsons.com.sg/)
5. pairwise function https://stackoverflow.com/questions/34562261/get-pairwise-iterator-with-additional-item-in-the-end (https://stackoverflow.com/questions/34562261/get-pairwise-iterator-with-additional-item-in-the-end)
6. distance function https://stackoverflow.com/questions/42686300/how-to-check-if-coordinate-inside-certain-area-python (https://stackoverflow.com/questions/42686300/how-to-check-if-coordinate-inside-certain-area-python)
7. Guide To Online Pet Store & Pet Shop Directory In Singapore (https://www.clubpets.com.sg/distribution-pet-shop/). This website provide a list of current pet store in Singapore.
8. Singapore government data set List of Licensed Vet Centres (https://data.gov.sg/dataset/list-of-licensed-vet-centres). (File name: list-of-licensed-vet-centres.zip)