DP0701EN-Week5-PeerGraded Project-report

February 25, 2020

1 DP0701EN-Week5-PeerGraded Project - Report

In this week, you will continue working on your capstone project. Please remember by the end of this week, you will need to submit the following:

1. A full report consisting of all of the following components (15 marks):

- Introduction where you discuss the business problem and who would be interested in this project.
- Data where you describe the data that will be used to solve the problem and the source of the data.
- Methodology section which represents the main component of the report where you discuss and describe any exploratory data analysis that you did, any inferential statistical testing that you performed, if any, and what machine learnings were used and why.
- Results section where you discuss the results.
- Discussion section where you discuss any observations you noted and any recommendations you can make based on the results.
- Conclusion section where you conclude the report.

2. A link to your Notebook on your Github repository pushed showing your code. (15 marks)

3. Your choice of a presentation or blogpost. (10 marks)

1.1 Topic:

1.2 The Battle of Neighborhoods - Where should I open a sushi restaurant in Toronto? - Part 2

1.2.1 Introduction

For a city of enthusiastic eaters, it's no surprise that food startups are emerging as one of Toronto's most attractive areas for entrepreneurship.

Toronto is a city where people form long lines for a taste of new restaurants. With a half of the city's population born outside of Canada, the downtown core and attached neighbourhoods easily offer anything from pho and jerk chicken to ballotines and samosas.

So Torontonians are always going to be interested in trying food from all over the world and Sushi is always among the Torontonians' favorite foods. In this report, I would like to explore the neighborhoods of the Toronto city and find out the potential best place to open a sushi restaurant.

1.2.2 Data

The data containing geolocations of sushi restaurants and neighborhoods will be collected from FourSquare. The location dataframe scraped from the online webpage consists of columns: * "PostalCode", * "Borough", * "Neighborhood", * "Latitude", * "Longitude".

The nearby-venue dataframe from FourSquare consists of columns: * "Neighborhood", * "Neighborhood Latitude", * "Neighborhood Longitude", * "Venue", * "Venue Latitude", * "Venue Longitude", * "Venue Category".

After counting by "Neighborhood", we will be able to list the top 5 most common venues in each neighborhood and the location of already existing sushi restaurants. In the end, we can carry out an unsupervised machine learning technique to determine the area of influence of the existing sushi restaurants, and find the possible optimal locations for our sushi start up.

Examples of Data-1

PostalCode	Borough	Neighborhood	Latitude	Longitude
AAA	XXXXXXXX	XXXXXXX	99.9999999	11.1111111

Examples of Data-2

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
XXXXXXXX	99.9999999	11.1111111	XXXXXXXX	99.9999999	11.1111111	XXXXXXXX

1.2.3 Methodology

We will preprocess the data from FourSquare and get the dataframe we introduced in the part "Data" together with some visualization of the data. Then we use an unsupervised machine learning technique to find the solution, i.e potential best neighborhood for our sushi start up.

Step 0: get ready

import libraries

```
[ ]: #!conda install -c conda-forge folium=0.5.0 --yes
     import numpy as np # library to handle data in a vectorized manner
     import pandas as pd # library for data analsysis
     pd.set_option('display.max_columns', None)
     pd.set_option('display.max_rows', None)

     import json # library to handle JSON files

     #!conda install -c conda-forge geopy --yes
     from geopy.geocoders import Nominatim # convert an address into latitude and␣
      ↪longitude values

     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 as mpl
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import matplotlib.colors as colors

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

#!conda install -c conda-forge folium=0.5.0 --yes b
import folium # map rendering library
from folium import plugins

print('Libraries imported.')
```

FourSquare ID and Credential

```
[ ]: CLIENT_ID = 'FDGPCM2RQRFB025MFXJVNTGPWASHXNOHERNUFARTZGK5IMFB' # your␣
     ↪Foursquare ID
     CLIENT_SECRET = 'JGNW3512LLQVHL1GDKTA205WUCKQV2DNW1I3FGCT1KQPUZ0T' # your␣
     ↪Foursquare Secret
     VERSION = '20180605' # Foursquare API version
```

define functions to be used later

```
[ ]: def return_most_common_venues(row, num_top_venues):
         row_categories = row.iloc[1:]
         row_categories_sorted = row_categories.sort_values(ascending=False)

         return row_categories_sorted.index.values[0:num_top_venues]
```

```
[ ]: def get_category_type(row):
         try:
             categories_list = row['categories']
         except:
             categories_list = row['venue.categories']

         if len(categories_list) == 0:
             return None
         else:
             return categories_list[0]['name']
```

```
[ ]: def getNearbyVenues(names, latitudes, longitudes, radius=500):
    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(name)

        # 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"]['groups'][0]['items']

        # return only relevant information for each nearby venue
        venues_list.append([(
            name,
            lat,
            lng,
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item␣
↪in venue_list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

```
[ ]: def add_markers(df, maptoadd):
         for (j, row) in df.iterrows():
             label = folium.Popup(row["Name"], parse_html = True)
             folium.CircleMarker(
                 [row["Latitude"], row["Longitude"]],
                 radius = 5,
                 popup = label,
                 color = 'red',
                 fill = True,
                 fill_color = '#3186cc',
                 fill_opacity = 0.7,
                 parse_html = False).add_to(maptoadd)
```

Step 1: Prepare the dataframe

Step 1.1: import data with neighborhood latitude and longitude

```
[ ]: df = pd.read_csv("Week3project.csv")
     Toronto = df[df['Borough'].str.contains("Toronto")]
     Toronto.reset_index(inplace = True)
     Toronto.drop(['index', 'Unnamed: 0'],axis = 1, inplace = True)
     Toronto.head()
```

Step 1.2: get coordinates

```
[ ]: address = 'Toronto'
     geolocator = Nominatim()
     location = geolocator.geocode(address)
     latitude = location.latitude
     longitude = location.longitude
     print('The geographical coordinate of Toronto are {}, {}.'.
      ↪format(latitude,longitude))
```

Step 1.3: visualization of neighborhoods

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

     for lat, lng, label in zip(Toronto['Latitude'], Toronto['Longitude'],␣
      ↪Toronto['Neighborhood']):
         label = folium.Popup(label, parse_html = True)
         folium.CircleMarker(
             [lat, lng],
             radius = 5,
             popup = label,
             color = 'blue',
             fill = True,
             fill_color = '#3186cc',
             fill_opacity = 0.7,
             parse_html = False).add_to(map_trt)

     map_trt
```

Step 1.4: get the top 500 venues that are in Toronto within a radius of 1000 meters

```
[ ]: LIMIT = 500 # limit of number of venues returned by Foursquare API

     radius = 1000 # define radius

     # create URL
     url = 'https://api.foursquare.com/v2/venues/explore?
      ↪&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
         CLIENT_ID,
         CLIENT_SECRET,
         VERSION,
         latitude,
         longitude,
         radius,
         LIMIT)
```

```
[ ]: results = requests.get(url).json()
```

Step 1.5: get information of nearby venues

```
[ ]: venues = results['response']['groups'][0]['items']
     nearby_venues = json_normalize(venues) # flatten JSON

     # filter columns
     filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat',␣
      ↪'venue.location.lng']
     nearby_venues = nearby_venues.loc[:, filtered_columns]

     # filter the category for each row
     nearby_venues['venue.categories'] = nearby_venues.apply(get_category_type, axis␣
      ↪= 1)

     # clean columns
     nearby_venues.columns = [col.split(".")[-1] for col in nearby_venues.columns]

     nearby_venues.head()
```

```
[ ]: venues_trt = getNearbyVenues(names = Toronto['Neighborhood'],
                                  latitudes = Toronto['Latitude'],
                                  longitudes = Toronto['Longitude'])
```

Step 2: Analyze Each Neighborhood

Step 2.1: prepare the dataframe

```
[ ]: # one hot encoding
     trt_onehot = pd.get_dummies(venues_trt[['Venue Category']], prefix = "",␣
      ↪prefix_sep = "")
     trt_onehot['Neighborhood'] = venues_trt['Neighborhood']
```

```
fixed_columns = [trt_onehot.columns[-1]] + list(trt_onehot.columns[:-1])
trt_onehot = trt_onehot[fixed_columns]
trt_grouped = trt_onehot.groupby('Neighborhood').mean().reset_index()
```

Step 2.1: print each neighborhood along with the top 10 most common venues

```
[ ]: num_top_venues = 10

     for hood in trt_grouped['Neighborhood']:
         print("----" + hood + "----")
         temp = trt_grouped[trt_grouped['Neighborhood'] == hood].T.reset_index()
         temp.columns = ['venue','freq']
         temp = temp.iloc[1:]
         temp['freq'] = temp['freq'].astype(float)
         temp = temp.round({'freq': 2})
         print(temp.sort_values('freq', ascending = False).reset_index(drop = True).
      ↪head(num_top_venues))
         print('\n')
```

Step 2.2: put the info in last step into a dataframe

```
[ ]: num_top_venues = 10

     indicators = ['st', 'nd', 'rd']

     # create columns according to number of top venues
     columns = ['Neighborhood']
     for ind in np.arange(num_top_venues):
         try:
             columns.append('{}{} Most Common Venue'.format(ind + 1,␣
      ↪indicators[ind]))
         except:
             columns.append('{}th Most Common Venue'.format(ind + 1))

     # create a new dataframe
     neighborhoods_venues_sorted = pd.DataFrame(columns = columns)
     neighborhoods_venues_sorted['Neighborhood'] = trt_grouped['Neighborhood']

     for ind in np.arange(trt_grouped.shape[0]):
         neighborhoods_venues_sorted.iloc[ind, 1:] =␣
      ↪return_most_common_venues(trt_grouped.iloc[ind, :], num_top_venues)

     neighborhoods_venues_sorted.head()
```

Step 2.3: visualize the most common venues

[ ]:
```
FirstVenue = pd.DataFrame(neighborhoods_venues_sorted["1st Most Common Venue"].
 ↪value_counts())
SecondVenue = pd.DataFrame(neighborhoods_venues_sorted["2nd Most Common Venue"].
 ↪value_counts())
ThirdVenue = pd.DataFrame(neighborhoods_venues_sorted["3rd Most Common Venue"].
 ↪value_counts())
FourthVenue = pd.DataFrame(neighborhoods_venues_sorted["4th Most Common Venue"].
 ↪value_counts())
FifthVenue = pd.DataFrame(neighborhoods_venues_sorted["5th Most Common Venue"].
 ↪value_counts())
```

[ ]:
```
FirstVenue.plot(kind = "barh", figsize = (8,5))
plt.title("Histogram of the 1st Most Common Venue in Toronto Neighborhoods")
plt.xlabel("Number")
plt.ylabel("Type")
plt.show()
```

[ ]:
```
SecondVenue.plot(kind = "barh", figsize = (8,5))
plt.title("Histogram of the 2nd Most Common Venue in Toronto Neighborhoods")
plt.xlabel("Number")
plt.ylabel("Type")
plt.show()
```

[ ]:
```
ThirdVenue.plot(kind = "barh", figsize = (8,5))
plt.title("Histogram of the 3rd Most Common Venue in Toronto Neighborhoods")
plt.xlabel("Number")
plt.ylabel("Type")
plt.show()
```

[ ]:
```
FourthVenue.plot(kind = "barh", figsize = (8,5))
plt.title("Histogram of the 4th Most Common Venue in Toronto Neighborhoods")
plt.xlabel("Number")
plt.ylabel("Type")
plt.show()
```

[ ]:
```
FifthVenue.plot(kind = "barh", figsize = (8,5))
plt.title("Histogram of the 5th Most Common Venue in Toronto Neighborhoods")
plt.xlabel("Number")
plt.ylabel("Type")
plt.show()
```

[ ]:
```
commonvenue = pd.merge(FirstVenue, SecondVenue, how = "outer", on = None,␣
 ↪left_index = True, right_index = True)
commonvenue = pd.merge(commonvenue, ThirdVenue, how = "outer", on = None,␣
 ↪left_index = True, right_index = True)
```

```
commonvenue = pd.merge(commonvenue, FourthVenue, how = "outer", on = None,␣
↪left_index = True, right_index = True)
commonvenue = pd.merge(commonvenue, FifthVenue, how = "outer", on = None,␣
↪left_index = True, right_index = True)

commonvenue.plot(kind = "bar", figsize = (20,8))
plt.title("Histogram of the 1st-5th Most Common Venue in Toronto Neighborhoods")
plt.xlabel("Number")
plt.ylabel("Type")
plt.show()
```

[ ]:
```
commonvenue.loc["Sushi Restaurant",].plot(kind = "bar", figsize = (10,5))
plt.title("Details of the Popularity of the Sushi Resaurant in Toronto␣
↪Neighborhoods")
plt.ylabel("Number")
plt.show()
```

Step 3: Find the best place for a sushi restaurant

Step 3.1: extract the sushi-related restaurants

[ ]:
```
sushi = []
for venue in venues:
    venuename = venue["venue"]
    categories = venue["venue"]["categories"]
    name = venue["venue"]["name"]
    location = venue["venue"]["location"]
    pluralname = categories[0]["pluralName"]
    address = location["formattedAddress"]
    lat = location["lat"]
    lng = location["lng"]

    if ("sushi" in pluralname.lower()) or ("japan" in pluralname.lower()) \
    or ("asia" in pluralname.lower())  or ("ramen" in pluralname.lower()):
        row = (name, address, lat, lng)
        sushi.append(row)

sushi = pd.DataFrame(sushi, columns=["Name", "Address", "Latitude",␣
↪"Longitude"])
sushi
```

Step 3.2: heatmap of the most common sushi-related restaurants

[ ]:
```
address = 'Toronto'
geolocator = Nominatim()
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
```

```
map_trt = folium.Map(location=[latitude, longitude], zoom_start = 15)

add_markers(sushi, map_trt)
heat = sushi[["Latitude", "Longitude"]].as_matrix().tolist()
map_trt.add_child(plugins.HeatMap(heat))

map_trt
```

Step 3.3: get the result of the potential best location

[ ]:
```
ourbest_lat = sushi["Latitude"].mean()
ourbest_lng = sushi["Longitude"].mean()
```

[ ]:
```
folium.CircleMarker(
    [ourbest_lat, ourbest_lng],
    radius = 8,
    popup = "Our Best Location",
    color = 'red',
    fill = True,
    fill_color = '#FF0000',
    fill_opacity = 0.7,
    parse_html = False).add_to(map_trt)

map_trt
```

1.2.4 Result and Conclusion

From taking the mean of most popular sushi-related restaurants in Toronto, we finally get our result of potential best place for a new sushi restaurant – On University Avenue to the south of University/Armoury.

1.2.5 Discussion

The sushi restaurant is among one of the most popular venues in the city of Toronto. With 1 3rd most common, 2 4th most common and 1 5th most common ranks in the neighborhoods of Toronto, we can see that opening a sushi restaurant can be a promising business to start.

Although there can be more factors affecting the choice of location in the real life such as rent, prices and traffic conditions, we are able to obtain the result of a potential best place by assessing the location and popularity of other popular sushi-related restaurants.

Finally, we have our best place - University Avenue to the south of University/Armoury. The address is in the neighborhood where sushi is loved and it also has a fair distance from all the other popular sushi-related restaurants.

[ ]:
```

```