Segmenting and clustering neighborhoods in Toronto assignment

March 22, 2019

```
In [2]: # importing necessary libraries
        import pandas as pd
        import numpy as np
        from bs4 import BeautifulSoup
        import requests
```

```
In [3]: # getting data from internet
        wikipedia_link='https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M'
        raw_wikipedia_page= requests.get(wikipedia_link).text

        # using beautiful soup to parse the HTML/XML codes.
        soup = BeautifulSoup(raw_wikipedia_page,'xml')
        #print(soup.prettify())
```

```
In [4]: # extracting the raw table inside that webpage
        table = soup.find('table')

        Postcode      = []
        Borough       = []
        Neighbourhood = []

        # print(table)

        # extracting a clean form of the table
        for tr_cell in table.find_all('tr'):

            counter = 1
            Postcode_var      = -1
            Borough_var       = -1
            Neighbourhood_var = -1

            for td_cell in tr_cell.find_all('td'):
                if counter == 1:
                    Postcode_var = td_cell.text
                if counter == 2:
                    Borough_var = td_cell.text
            tag_a_Borough = td_cell.find('a')

        if counter == 3:
            Neighbourhood_var = str(td_cell.text).strip()
            tag_a_Neighbourhood = td_cell.find('a')

        counter +=1

    if (Postcode_var == 'Not assigned' or Borough_var == 'Not assigned' or Neighbourhood
        continue
    try:
        if ((tag_a_Borough is None) or (tag_a_Neighbourhood is None)):
            continue
    except:
        pass
    if(Postcode_var == -1 or Borough_var == -1 or Neighbourhood_var == -1):
        continue

    Postcode.append(Postcode_var)
    Borough.append(Borough_var)
    Neighbourhood.append(Neighbourhood_var)
```

```
In [10]: unique_p = set(Postcode)
         print('num of unique Postal codes:', len(unique_p))
         Postcode_u      = []
         Borough_u       = []
         Neighbourhood_u = []


         for postcode_unique_element in unique_p:
             p_var = ''; b_var = ''; n_var = '';
             for postcode_idx, postcode_element in enumerate(Postcode):
                 if postcode_unique_element == postcode_element:
                     p_var = postcode_element;
                     b_var = Borough[postcode_idx]
                     if n_var == '':
                         n_var = Neighbourhood[postcode_idx]
                     else:
                         n_var = n_var + ', ' + Neighbourhood[postcode_idx]
             Postcode_u.append(p_var)
             Borough_u.append(b_var)
             Neighbourhood_u.append(n_var)
```

```
num of unique Postal codes: 84
```

```
In [9]: toronto_dict = {'Postcode':Postcode_u, 'Borough':Borough_u, 'Neighbourhood':Neighbourhoo
        df_toronto = pd.DataFrame.from_dict(toronto_dict)
        df_toronto.to_csv('toronto_part1.csv')
        df_toronto.head(14)

Out[9]:    Postcode           Borough                    Neighbourhood
        0       M9A         Etobicoke                 Islington Avenue
        1       M4H         East York                 Thorncliffe Park
        2       M1B       Scarborough                   Rouge, Malvern
        3       M9W         Etobicoke                        Northwest
        4       M9L        North York                    Humber Summit
        5       M4Y  Downtown Toronto             Church and Wellesley
        6       M9N              York                           Weston
        7       M3J        North York  Northwood Park, York University
        8       M2H        North York                Hillcrest Village
        9       M2J        North York                       Henry Farm
        10      M5S  Downtown Toronto            University of Toronto
        11      M1T       Scarborough                    Tam O'Shanter
        12      M6L        North York                  Maple Leaf Park
        13      M1W       Scarborough                     Steeles West

In [8]: df_toronto.shape

Out[8]: (84, 3)

In [ ]:
```