Data Science Capstone Week 3 Assignment - PART 1

Intent: Scrape a Wikipedia page and extract a pandas dataframe containing Toronto Postalcodes, Borough, and Neighborhood

Import Pandas Library

...to provide the Dataframe container as well as the scraper "read_html" command

In [242]:

```
import pandas as pd
import numpy as np
```

Import Urllib Library

...to read the HTML code into a string variable "jenhtml" ... this will facilitate the execution of pandas.read_html the scaping command

In [243]:

```
import urllib.request
import urllib.parse
import urllib.error

!conda install -c conda-forge lxml --yes
import lxml

print("packages imported")
```

```
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.5.11
  latest version: 4.7.11

Please update conda by running

    $ conda update -n base -c defaults conda



# All requested packages already installed.

packages imported
```

Open and Read Wikipedia Page into a String Variable

...to read the HTML code into a string variable "jenhtml" ... this will facilitate the execution of pandas.read_html the scraping command

In [244]:

```
jenurl = "https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M"
print(jenurl)
```

```
https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M
```

In [245]:

```
jenhtmlStr = urllib.request.urlopen(jenurl).read()
```

In [246]:

```
jendf = pd.read_html(jenhtmlStr)[0]
jendf.head()
```

Out[246]:

	Postcode	Borough	Neighbourhood
0	M1A	Not assigned	Not assigned
1	M2A	Not assigned	Not assigned
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront

In [247]:

```
jendf.shape
```

Out[247]:

```
(288, 3)
```

Drop Rows with Missing Boroughs

In [248]:

```
jendf = jendf[jendf.Borough != "Not assigned"]
jendf.head()
```

Out[248]:

	Postcode	Borough	Neighbourhood
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront
5	M5A	Downtown Toronto	Regent Park
6	M6A	North York	Lawrence Heights

Combine Rows of Postcode-sharing Neighborhoods

- determine which postcodes have more than one neighborhood
- for each multiply-used postal code:

Replace all multiply-used rows of the same postal code with just one row.

The new value for the "Neighbourhood" field will be a list of the former neighborhoods that appeared in the individual rows corresponding to the shared zipcode.

Find the multiply-used Postcodes by performing a df.value_counts()

In [249]:

```
mycounts = jendf["Postcode"].value_counts()
mycounts.head()
```

Out[249]:

```
M8Y    8
M9V    8
M5V    7
M9B    5
M8Z    5
Name: Postcode, dtype: int64
```

Form a LIST of Postcodes that have more than one neighborhood

In [250]:

```
postcodeList = mycounts.index.values.tolist()
pcodeList = []
for pcode in postcodeList:
    if mycounts.loc[pcode] > 1:
        pcodeList.append(str(pcode))

print(len(pcodeList))
neighList = []
boroList = []
```

```
57
```

Update the Dataframe by deleting duplications and augmenting 'Neighborhood' if necessary

The algorithm employed is:

For each postal code "pc" in pcodeList (created above):

1. **Select and extract all rows in main dataframe "jendf" with postalcode "pc" and place into another temporary dataframe called 'tempDF'**
2. **Form a new neighborhood String by extracting all individual neighborhoods in tempDF, putting them in a list, and invoking "join" with comma as delimiter**
3. **Save the new neighborhood string 'nStr' in a list; save its borough in a list called "boroList"**

After the loop, save all collected information in a dictionary: "nndict"

In [251]:

```
#initialize lists
boroList = []
neighList = []
nList = list()

# Traverse pcodeList to create the new 'Neighbourhood' strings
for pc in pcodeList:
    nList = []
    # 1. select and extract all rows in main dataframe "jendf" with
    #    Postalcode "pc" and place into another temporary dataframe 'tempDF'
    tempdfSeries = jendf.Postcode == pc
    tempDF = jendf[tempdfSeries]
    # ----------------------------------------
    # 2a.  Form a new neighborhood String by extracting all individual neighborhood
s
    #     in tempDF and putting them in a list
    nList = tempDF['Neighbourhood'].values.tolist()
    # ----------------------------------------
    # 2b. invoke python function 'join()' with comma as delimiter
    jenDelimiter = ', '
    nStr = jenDelimiter.join(nList)
    # ----------------------------------------
    #3.   Save the new neighborhood string 'nStr' in a list; save its borough in a
 list
    neighList.append(nStr)
    jenBoro = str(tempDF.iloc[0,1])
    boroList.append(jenBoro)
## end FOR LOOP

# Store all the collected information in a "New Neighbourhood" dictionary
nnDict = dict()

nnDict['Postcode'] = pcodeList
nnDict['Borough'] = boroList
nnDict['Neighbourhood'] = neighList
```

Remove Redundancy in Dataframe by deleting multiple rows and finally replacing 'Neighborhood' if necessary

The algorithm employed is:

A. Delete all postalcode duplicates from our original DF called "jendf"

B. Create a new pandas dataframe "newNeighDF" to concatenate with our original DF "jendf"

C. Merge the new dataframe "newNeighDF" into original "jendf" by using the pandas "concat" command

A. Delete all postalcode duplicates

In [252]:

```
# A.    Delete all postalcode duplicates
jendf.drop_duplicates('Postcode', keep=False, inplace=True)
```

B. Create a new pandas dataframe "newNeighDF" to concatenate with our original DF "jendf"

In [254]:

```
newNeighDF =pd.DataFrame(data=nnDict)
newNeighDF.head()
```

Out[254]:

	Postcode	Borough	Neighbourhood
0	M8Y	Etobicoke	Humber Bay, King's Mill Park, Kingsway Park So...
1	M9V	Etobicoke	Albion Gardens, Beaumond Heights, Humbergate, ...
2	M5V	Downtown Toronto	CN Tower, Bathurst Quay, Island airport, Harbo...
3	M9B	Etobicoke	Cloverdale, Islington, Martin Grove, Princess ...
4	M8Z	Etobicoke	Kingsway Park South West, Mimico NW, The Queen...

C. Merge the new dataframe "newNeighDF" into original "jendf" by using the pandas "concat" command

In [256]:

```
jenFrames = [jendf, newNeighDF]
postalDF = pd.concat(jenFrames)
postalDF.reset_index(drop=True,inplace=True)
postalDF.head(20)
```

Out[256]:

	Postcode	Borough	Neighbourhood
0	M3A	North York	Parkwoods
1	M4A	North York	Victoria Village
2	M7A	Queen's Park	Not assigned
3	M9A	Etobicoke	Islington Avenue
4	M3B	North York	Don Mills North
5	M6B	North York	Glencairn
6	M4C	East York	Woodbine Heights
7	M5C	Downtown Toronto	St. James Town
8	M6C	York	Humewood-Cedarvale
9	M4E	East Toronto	The Beaches
10	M5E	Downtown Toronto	Berczy Park
11	M6E	York	Caledonia-Fairbanks
12	M1G	Scarborough	Woburn
13	M4G	East York	Leaside
14	M5G	Downtown Toronto	Central Bay Street
15	M6G	Downtown Toronto	Christie
16	M1H	Scarborough	Cedarbrae
17	M2H	North York	Hillcrest Village
18	M4H	East York	Thorncliffe Park
19	M1J	Scarborough	Scarborough Village

Replace "Not assigned" Neighbourhood values

Employ the following algorithm:

- **(1) Extract all rows in dataframe postalDF with Neighbourhood = 'Not assigned' into a new dataframe naNeighDF**
- **(2) In dataframe naNeighDF assign "Neighbourhood" column to be equal to "Borough" column**
- **(3) Delete all rows from original dataframe postalDF with Neighbourhood = 'Not assigned'**
- **(4) Concatenate the two dataframes postalDF and naNeighDF into regionsTorontoDF**

In [257]:

```
# (1) Extract all rows in dataframe 'newNeighDF' with Neighbourhood = 'Not assigned' into a new dataframe naNeighDF
naNeighDF = postalDF[postalDF.Neighbourhood=='Not assigned']
naNeighDF
```

Out[257]:

	Postcode	Borough	Neighbourhood
2	M7A	Queen's Park	Not assigned

In [258]:

```
#(2) In dataframe "naNeighDF" assign "Neighbourhood" column to be equal to "Borough" column
naNeighDF['Neighbourhood']=naNeighDF['Borough']
naNeighDF
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages/ipykerne
l_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-d
ocs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[258]:

	Postcode	Borough	Neighbourhood
2	M7A	Queen's Park	Queen's Park

In [259]:

```
naNeighDF.head(10)
```

Out[259]:

	Postcode	Borough	Neighbourhood
2	M7A	Queen's Park	Queen's Park

In [260]:

```
# (3) Delete all rows from original dataframe "postalDF" with "Neighbourhood = 'Not
assigned'"
postalDF = postalDF[postalDF.Neighbourhood!='Not assigned']
```

In [261]:

```
#(4) Concatenate the two dataframes <b>postalDF </b> and <b>naNeighDF</b> into regi
onsTorontoDF
myframes = [postalDF, naNeighDF]
regionsTorontoDF = pd.concat(myframes)
regionsTorontoDF.shape
```

Out[261]:

```
(103, 3)
```

In [262]:

```
regionsTorontoDF.reset_index(drop=True,inplace=True)
regionsTorontoDF.head(14)
```

Out[262]:

	Postcode	Borough	Neighbourhood
0	M3A	North York	Parkwoods
1	M4A	North York	Victoria Village
2	M9A	Etobicoke	Islington Avenue
3	M3B	North York	Don Mills North
4	M6B	North York	Glencairn
5	M4C	East York	Woodbine Heights
6	M5C	Downtown Toronto	St. James Town
7	M6C	York	Humewood-Cedarvale
8	M4E	East Toronto	The Beaches
9	M5E	Downtown Toronto	Berczy Park
10	M6E	York	Caledonia-Fairbanks
11	M1G	Scarborough	Woburn
12	M4G	East York	Leaside
13	M5G	Downtown Toronto	Central Bay Street

In [263]:

```
regionsTorontoDF.shape
```

Out[263]:

```
(103, 3)
```