Assignment Task 1.3.2

August 4, 2024

[7]:
```
import time
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from bs4 import BeautifulSoup
import pandas as pd

# Correct path to the Chrome browser executable
chrome_path = "C:\Program Files\Google\Chrome\Application\chrome.exe"

# Set up Selenium WebDriver for Chrome
options = Options()
options.binary_location = chrome_path

# Correct path to the ChromeDriver executable
driver_path = 'C:\\JN\\chromedriver.exe'
# Initialize the Service object with the driver path
service = Service(executable_path=driver_path)
driver = webdriver.Chrome(service=service, options=options)

# URL of the webpage containing the county names
url = 'https://www.federalreserve.gov/releases/z1/dataviz/household_debt/county/
 ↪table/'

# Fetching the webpage content using Selenium
driver.get(url)


time.sleep(5)
# Get the page source and parse it with BeautifulSoup
soup = BeautifulSoup(driver.page_source, 'html.parser')
driver.quit()

# Find the table containing the county names
table = soup.find('table')

# Check if the table is found
```

```
if table is not None:
    print("Table found. Proceeding to extract data.")

    # Extract county names from the third column of the table
    county_names = []
    rows = table.find_all('tr')

    # Checking if rows are found
    if len(rows) > 1:
        print("Rows found in the table. Extracting county names.")

        for row in rows[1:]:  # Skip the header row
            cols = row.find_all('td')

            # Ensuring there are enough columns in the row
            if len(cols) > 2:
                county_name = cols[2].text.strip()
                county_names.append(county_name)
            else:
                print("Row does not have enough columns. Skipping row.")

        # Converting to DataFrame
        county_df = pd.DataFrame(county_names, columns=['County Name'])

        # Display the number of county names extracted
        print(f"Number of county names extracted: {len(county_names)}")
    else:
        print("No rows found in the table.")
else:
    print("Table not found on the webpage.")

# Load the existing CSV file
file_path = 'C:
 ↪\\Users\\THINKPAD\\Downloads\\household-debt-by-county\\household-debt-by-county.
 ↪csv'  # Backend Data
df = pd.read_csv(file_path)

# Droping the 'quarter' column
df = df.drop(columns=['qtr'])

# Combine the 'Lower bound' and 'Upper bound' columns into a new column 'dti'
df['dti'] = df['low'].astype(str) + '-' + df['high'].astype(str)

# Drop the original 'Lower bound' and 'Upper bound' columns
df = df.drop(columns=['low', 'high'])

# Display the number of rows in the dataset
```

```
print(f"Number of rows in the dataset: {len(df)}")

# Creating a mapping of unique counties to their names
county_mapping = dict(enumerate(county_df['County Name']))

# Calculating how many times each county should be repeated
repeat_factor = len(df) // len(county_mapping)

# Creating a new column 'County Name' in the original dataset
df['County Name'] = df.index.map(lambda x: county_mapping[x %␣
 ↪len(county_mapping)])

# Display some information
print(f"Number of unique counties: {len(county_mapping)}")
print(f"Number of rows per county: {repeat_factor}")

# Save the merged data to a new CSV file
output_path = 'C:\\JN\\dti_data_with_county_names.csv'
df.to_csv(output_path, index=False)
print(f"Data saved to {output_path}")
print("Data extraction, merging, and formatting complete.")

# to verify the result
print(df.head())
```

```
Table found. Proceeding to extract data.
Rows found in the table. Extracting county names.
Number of county names extracted: 3137
Number of rows in the dataset: 313819
Number of unique counties: 3137
Number of rows per county: 100
Data saved to C:\JN\dti_data_with_county_names.csv
Data extraction, merging, and formatting complete.
   year  area_fips        dti  County Name
0  1999       1001  1.82-2.15  Autauga, AL
1  1999       1003  1.82-2.15  Baldwin, AL
2  1999       1005   0.0-0.78  Barbour, AL
3  1999       1007  2.61-3.43     Bibb, AL
4  1999       1009  1.82-2.15   Blount, AL
```

[ ]: