datafetch

October 8, 2024

0.0.1 Data Source

This project utilizes data from the **National Survey on Drug Use and Health, 2015 (NSDUH-2015)**, provided by the Substance Abuse and Mental Health Services Administration (SAMHSA). The dataset contains detailed information on drug use, mental health, and related factors among individuals in the United States for the year 2015.

- **Dataset**: National Survey on Drug Use and Health, 2015 (NSDUH-2015)
- **Codebook (Description of Columns)**: NSDUH 2015 Codebook

[ ]:
```
import pandas as pd
import os
from typing import List, Optional, Dict
import gc
```

[2]:
```
# !pip3 install pandas
# !pip3 install pyarrow
```

0.0.2 Data Fetch

[3]:
```
def fetch_nsduh_data(year: int) -> Optional[pd.DataFrame]:
    """
    Fetches NSDUH data for a specified year from a remote source.

    Args:
        year (int): The year for which to fetch data.

    Returns:
        Optional[pd.DataFrame]: A pandas DataFrame with the data, or None if␣
↪fetching failed.
    """
    url_placeholder = "https://www.datafiles.samhsa.gov/sites/default/files/
↪field-uploads-protected/studies/NSDUH-{year}/NSDUH-{year}-datasets/
↪NSDUH-{year}-DS0001/NSDUH-{year}-DS0001-bundles-with-study-info/
↪NSDUH-{year}-DS0001-bndl-data-tsv.zip"

    try:
        url = url_placeholder.format(year=year)
        df = pd.read_csv(url, compression='zip', sep='\t', low_memory=False)
        return df
    except Exception as e:
        print(f"An error occurred for year {year}: {e}")
        return None
```

[4]:
```
def check_parquet_exists(years: List[int], output_dir: str) -> Dict[int, bool]:
    """
    Checks if Parquet files for the specified years already exist in the output␣
↪directory.

    Args:
        years (List[int]): A list of years to check.
        output_dir (str): The directory where Parquet files are saved.

    Returns:
        Dict[int, bool]: A dictionary with years as keys and boolean values␣
↪indicating
                         whether the Parquet file for that year exists.
    """
    existence_check = {}

    for year in years:
        year_path = os.path.join(output_dir, f'year={year}')
        existence_check[year] = os.path.exists(year_path)

    return existence_check
```

[5]:
```
def write_parquet(df: pd.DataFrame, year: int, output_dir: str, overwrite: bool␣
↪= False) -> None:
    """
    Writes a DataFrame to Parquet format, partitioned by year.

    Args:
        df (pd.DataFrame): The DataFrame to write.
        year (int): The year of the data.
        output_dir (str): The directory where Parquet files will be saved.
        overwrite (bool): If True, overwrite existing files. If False, skip␣
↪existing files.
    """
    try:
        year_dir = os.path.join(output_dir, f'year={year}')
        if overwrite and os.path.exists(year_dir):
            shutil.rmtree(year_dir)  # Remove existing directory to start fresh

        df['year'] = year  # Add the year column for partitioning
        # Write data to Parquet format with partitioning
        df.to_parquet(output_dir, partition_cols=['year'], index=False)
        print(f"Data for year {year} successfully saved to Parquet format in␣
↪{output_dir}")
    except Exception as e:
        print(f"Error saving data to Parquet for year {year}: {e}")
```

[6]:
```
def data_fetch(years_to_fetch: List[int], output_dir: str, overwrite: bool =␣
↪False) -> None:
    """
    Fetches NSDUH data for specified years and saves it to Parquet format, one␣
↪year at a time.

    Args:
        years_to_fetch (List[int]): A list of years for which to fetch data.
        output_dir (str): The directory where Parquet files will be saved.
        overwrite (bool): If True, overwrite existing files. If False, skip␣
↪existing files.
    """
    try:
        for year in years_to_fetch:
            if not overwrite and os.path.exists(os.path.join(output_dir,␣
↪f'year={year}')):
                print(f"Data for year {year} already exists. Skipping.")
                continue

            df = fetch_nsduh_data(year)
            if df is not None:
                print(f"Successfully fetched data for year: {year}")
                write_parquet(df, year, output_dir, overwrite)
                del df  # Remove the DataFrame from memory
                gc.collect()  # Force garbage collection

        print("All requested years processed.")
    except Exception as e:
        print(f"An unexpected error occurred in the data_fetch function: {e}")
```

[7]:
```
def read_parquet(input_dir: str, years: Optional[List[int]] = None) ->␣
↪Dict[int, pd.DataFrame]:
    """
    Reads Parquet files for specified years from the input directory.

    Args:
        input_dir (str): The directory where Parquet files are stored.
        years (Optional[List[int]]): A list of years to read. If None, read all␣
↪available years.

```

```
    Returns:
        Dict[int, pd.DataFrame]: A dictionary with years as keys and pandas␣
↪DataFrames as values.
    """
    data_frames = {}
    available_years = [int(d.split('=')[1]) for d in os.listdir(input_dir) if d.
↪startswith('year=')]
    years_to_read = years if years is not None else available_years

    for year in years_to_read:
        year_path = os.path.join(input_dir, f'year={year}')
        if os.path.exists(year_path):
            df = pd.read_parquet(year_path)
            data_frames[year] = df
        else:
            print(f"Warning: No data found for year {year}")

    return data_frames
```

[8]:
```
years = [2015, 2016, 2017, 2018, 2019]
output_directory = "../data/DS/NSDUH"

# Fetch and save data
data_fetch(years, output_directory, overwrite=False)

# Read saved data (if needed)
# Note: This part is optional and can be removed if you don't need to read the␣
↪data immediately after saving
for year in years:
    df = read_parquet(output_directory, [year])
    if year in df:
        print(f"Data for year {year}:")
        print(df[year].head())
    del df
    gc.collect()
```

```
Successfully fetched data for year: 2015
Data for year 2015 successfully saved to Parquet format in ../data/DS/NSDUH
Successfully fetched data for year: 2016
Data for year 2016 successfully saved to Parquet format in ../data/DS/NSDUH
Successfully fetched data for year: 2017
Data for year 2017 successfully saved to Parquet format in ../data/DS/NSDUH
Successfully fetched data for year: 2018
Data for year 2018 successfully saved to Parquet format in ../data/DS/NSDUH
Successfully fetched data for year: 2019
Data for year 2019 successfully saved to Parquet format in ../data/DS/NSDUH
All requested years processed.
```

```
Data for year 2015:
   QUESTID2    FILEDATE  CIGEVER  CIGOFRSM  CIGWILYR  CIGTRY  CIGYFU  CIGMFU  \
0  25095143  02/15/2018        1        99        99      16    2014       1
1  13005143  02/15/2018        1        99        99      15    9999      99
2  67415143  02/15/2018        2        99        99     991    9991      91
3  70925143  02/15/2018        2         3         4     991    9991      91
4  75235143  02/15/2018        1        99        99      17    9999      99

   CIGREC  CIG30USE  …  POVERTY3  TOOLONG  TROUBUND  PDEN10  COUTYP4  \
0       2        93  …       1.0        2         2       3        3
1       3        93  …       2.0        1         2       2        2
2      91        91  …       1.0        2         2       2        3
3      91        91  …       3.0        2         2       2        2
4       1        22  …       1.0        2         2       3        3

   MAIIN102  AIIND102     ANALWT_C  VESTR  VEREP
0         2         2  1088.413235  40028      1
1         2         2  4423.360328  40025      1
2         2         2   328.111801  40004      2
3         2         2   235.290450  40027      1
4         2         2  2280.878615  40011      1

[5 rows x 2679 columns]
Data for year 2016:
   QUESTID2    FILEDATE  CIGEVER  CIGOFRSM  CIGWILYR  CIGTRY  CIGYFU  CIGMFU  \
0  11635143  02/28/2018        1        99        99      16    9999      99
1  36845143  02/28/2018        1        99        99      15    9999      99
2  35755143  02/28/2018        1        99        99      26    9999      99
3  94475143  02/28/2018        2         4         4     991    9991      91
4  92675143  02/28/2018        1        99        99       5    9999      99

   CIGREC  CIG30USE  …  POVERTY3  TOOLONG  TROUBUND  PDEN10  COUTYP4  \
0       4        93  …       3.0        2         2       3        3
1       1         7  …       3.0        1         2       2        2
2       1         7  …       2.0        2         2       1        1
3      91        91  …       3.0        2         2       1        1
4       4        93  …       3.0        2         2       1        1

   MAIIN102  AIIND102      ANALWT_C  VESTR  VEREP
0         2         2    819.434247  40037      2
1         2         2    280.624352  40013      2
2         2         2  10133.833583  40036      2
3         2         2   2284.717175  40028      1
4         2         2  24815.892373  40009      2

[5 rows x 2668 columns]
Data for year 2017:
   QUESTID2    FILEDATE  CIGEVER  CIGOFRSM  CIGWILYR  CIGTRY  CIGYFU  CIGMFU  \
0  55235143  10/09/2018        1        99        99      13    9999      99
1  13435143  10/09/2018        1        99        99      15    9999      99
2  81345143  10/09/2018        1        99        99      14    9999      99
3  53955143  10/09/2018        1        99        99      16    9999      99
4  51775143  10/09/2018        2        99        99     991    9991      91

   CIGREC  CIG30USE  …  POVERTY3  TOOLONG  TROUBUND  PDEN10  COUTYP4  \
0       4        93  …       3.0        2         2       1        1
1       1        18  …       3.0        1         2       1        1
2       1        10  …       3.0        2         2       1        1
3       4        93  …       3.0        2         2       2        2
4      91        91  …       3.0        1         1       1        1

   MAIIN102  AIIND102      ANALWT_C  VESTR  VEREP
0         2         2  11203.888954  40043      1
1         2         2   9496.462244  40006      2
2         2         2   2943.702802  40030      2
3         2         2   1783.702549  40026      2
4         2         2  31528.749357  40029      1

[5 rows x 2668 columns]
Data for year 2018:
   QUESTID2    FILEDATE  CIGEVER  CIGOFRSM  CIGWILYR  CIGTRY  CIGYFU  CIGMFU  \
0  11015143  10/08/2019        1        99        99      12    9999      99
1  86325143  10/08/2019        2        99        99     991    9991      91
2  35425143  10/08/2019        1        99        99      13    9999      99
3  98125143  10/08/2019        2        99        99     991    9991      91
4  15945143  10/08/2019        2        99        99     991    9991      91

   CIGREC  CIG30USE  …  POVERTY3  TOOLONG  TROUBUND  PDEN10  COUTYP4  \
0       1        20  …       3.0        2         2       1        1
1      91        91  …       2.0        2         2       3        3
2       4        93  …       3.0        2         2       2        2
3      91        91  …       2.0        2         2       2        2
4      91        91  …       2.0        1         2       1        1

   MAIIN102  AIIND102      ANALWT_C  VESTR  VEREP
0         2         2  20783.261908  40001      1
1         2         2   1095.884074  40004      1
2         2         2    374.445005  40017      2
3         2         2   2421.263435  40038      2
4         2         2   6637.319591  40040      2

[5 rows x 2691 columns]
Data for year 2019:
   QUESTID2    FILEDATE  CIGEVER  CIGOFRSM  CIGWILYR  CIGTRY  CIGYFU  CIGMFU  \
0  43295143  10/09/2020        1        99        99      13    9999      99
1  65095143  10/09/2020        2        99        99     991    9991      91
```

```
2  49405143  10/09/2020        1        99        99      22    9999      99
3  51015143  10/09/2020        2        99        99     991    9991      91
4  31825143  10/09/2020        2        99        99     991    9991      91

   CIGREC  CIG30USE  …  POVERTY3  TOOLONG  TROUBUND  PDEN10  COUTYP4  \
0       4        93  …       3.0        2         2       2        2
1      91        91  …       3.0        2         2       2        2
2       4        93  …       3.0        2         2       2        2
3      91        91  …       1.0        2         2       2        2
4      91        91  …       3.0        2         2       2        2

   MAIIN102  AIIND102     ANALWT_C  VESTR  VEREP
0         2         2  6613.865847  40004      2
1         2         2  6321.580570  40003      1
2         2         2  5045.607492  40008      1
3         2         2  2419.558820  40031      1
4         2         2   575.225454  40010      2

[5 rows x 2741 columns]
```

0.0.3 Loading a second dataset: Study of Online Gaming and it's affect on Mental Health (Anxiety).

About this dataset

This dataset consists of data collected as a part of a survey among gamers worldwide. The questionnaire asked questions that psychologists generally ask people who are prone to anxiety, social phobia, and less to no life satisfaction. The questionnaire consists of several set of questions as asked as a part of psychological study. The original data was collated by Marian Sauter and Dejan Draschkow.

Kaggle dataset source: https://www.kaggle.com/datasets/divyansh22/online-gaming-anxiety-data

[3]:
```python
import requests, zipfile, io
import pandas as pd
# URL to direct download the dataset
# We will need to later modify this approach to use Kaggle API with␣
 ↪authentication key.
# Need to figure out to securely implement this approach - target for phase 2

url = 'https://storage.googleapis.com/kaggle-data-sets/820200/1403222/
 ↪compressed/GamingStudy_data.csv.zip?
 ↪X-Goog-Algorithm=GOOG4-RSA-SHA256&X-Goog-Credential=gcp-kaggle-com%40kaggle-161607.
 ↪iam.gserviceaccount.
 ↪goog4_request&X-Goog-Date=20241008T143813Z&X-Goog-Expires=259200&X-Goog-SignedHeaders=host&

# Download the file
response = requests.get(url)
```

```
# Check if the download was successful
if response.status_code == 200:
    # Create a file-like object from the response content
    zip_file = zipfile.ZipFile(io.BytesIO(response.content))

    # Iterate over the files in the zip archive
    for file_name in zip_file.namelist():
        with zip_file.open(file_name) as extracted_file:
            # Read the content of the extracted file
            gaming_dat = pd.read_csv(extracted_file, encoding='ISO-8859-1')
            gaming_dat.to_csv(r"../data/GamingStudy_data.csv", index=False)

    print('File downloaded and extracted successfully!')
else:
    print('Using the file stored locally in the folder named data')
    gaming_dat = pd.read_csv(r"./data/gamedata.csv", encoding='ISO-8859-1')

gaming_dat.head()
```

```
File downloaded and extracted successfully!
```

```
[3]:    S. No.    Timestamp  GAD1  GAD2  GAD3  GAD4  GAD5  GAD6  GAD7  \
     0       1  42052.00437     0     0     0     0     1     0     0
     1       2  42052.00680     1     2     2     2     0     1     0
     2       3  42052.03860     0     2     2     0     0     3     1
     3       4  42052.06804     0     0     0     0     0     0     0
     4       5  42052.08948     2     1     2     2     2     3     2

                      GADE  …  Birthplace    Residence  Reference  \
     0  Not difficult at all  …         USA          USA     Reddit
     1    Somewhat difficult  …         USA          USA     Reddit
     2  Not difficult at all  …     Germany      Germany     Reddit
     3  Not difficult at all  …         USA          USA     Reddit
     4        Very difficult  …         USA  South Korea     Reddit

                                              Playstyle  accept GAD_T SWL_T  \
     0                                         Singleplayer  Accept     1    23
     1              Multiplayer - online - with strangers  Accept     8    16
     2                                         Singleplayer  Accept     8    17
     3  Multiplayer - online - with online acquaintanc…  Accept     0    17
     4              Multiplayer - online - with strangers  Accept    14    14

        SPIN_T Residence_ISO3 Birthplace_ISO3
     0     5.0            USA             USA
     1    33.0            USA             USA
     2    31.0            DEU             DEU
     3    11.0            USA             USA
```

```
4    13.0            KOR              USA

[5 rows x 55 columns]
```

```
[ ]:
```