Coding Challenge for Data Engineering

Author: Sixue Liu
Contact: sixueliu@uchicago.edu (mailto:sixueliu@uchicago.edu)
Date: May 28, 2020

In [1]:
```
# Import Packages

import pandas as pd
import numpy as np
```

In [2]:
```
# Load data as dataframe

raw = pd.read_csv('raw.txt')
```

In [3]:
```
# Check for format

raw.head()
```

Out[3]:

	Year	Rank	Company	Revenue (in millions)	Profit (in millions)
0	1955	1	General Motors	9823.5	806
1	1955	2	Exxon Mobil	5661.4	584.8
2	1955	3	U.S. Steel	3250.4	195.4
3	1955	4	General Electric	2959.1	212.6
4	1955	5	Esmark	2510.8	19.1

In [4]:
```
# Simplify column names

raw.columns = ['Year', 'Rank', 'Company', 'Revenue', 'Profit']
```

1. Print out how many rows of the data is in the CSV data.

In [5]:
```
print('There are {} rows of the data in this CSV file'.format(len(raw)))
```

```
There are 25500 rows of the data in this CSV file
```

2. Remove all rows with 'profit' that is not numerical value. Then print out how many rows of data are left, after removal of the rows with invalid non-numeric profit column data.

In [6]:

```
# Check for data type

raw.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25500 entries, 0 to 25499
Data columns (total 5 columns):
Year        25500 non-null int64
Rank        25500 non-null int64
Company     25500 non-null object
Revenue     25500 non-null float64
Profit      25500 non-null object
dtypes: float64(1), int64(2), object(2)
memory usage: 996.2+ KB
```

In [7]:

```
# Checking the content of "Profit" column, the non-numeric value is represe

raw['Profit'].value_counts()
```

Out[7]:

```
N.A.       369
4           73
3           71
5.7         67
6           67
          ...
4735         1
475.3        1
1397         1
3235.9       1
-217         1
Name: Profit, Length: 6977, dtype: int64
```

In [8]:

```
# Count the number of non-numeric value in the "Profit" column

raw[raw['Profit'] == 'N.A.'].count()
```

Out[8]:

```
Year        369
Rank        369
Company     369
Revenue     369
Profit      369
dtype: int64
```

In [9]:

```
# Remove all rows with "Profit" that is not numerical value

dropped = raw[(raw['Profit'] == 'N.A.')].index
raw = raw.drop(dropped)
```

In [10]:

```
print('There are {} rows of the data left, after removal of the rows \
    with invalid non-numeric profit column data.'.format(len(raw)))
```

```
There are 25131 rows of the data left, after removal of the rows     with
invalid non-numeric profit column data.
```

In [11]:

```
# Change the data type of "Profit" column to float

raw = raw.astype({"Profit": float})
```

In [12]:

```
# Check for data type

raw.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 25131 entries, 0 to 25499
Data columns (total 5 columns):
Year       25131 non-null int64
Rank       25131 non-null int64
Company    25131 non-null object
Revenue    25131 non-null float64
Profit     25131 non-null float64
dtypes: float64(2), int64(2), object(1)
memory usage: 1.2+ MB
```

You can now convert the content into JSON format and write it out to another file called data2.json which should only contain rows of data that has valid profit values.

In [13]:

```
# Convert and export the JSON file

raw.to_json(r'data2.json')
```

3. Order the data based on the profit value. Print the top 20 rows with the highest profit values.

In [14]:

```
# Sort the data based on the "Profit"

raw = raw.sort_values(by='Profit', ascending=False)
```

In [15]:
```
# Print the top 20 rows with the highest profit values

raw[:20]
```

Out[15]:

	Year	Rank	Company	Revenue	Profit
25001	2005	2	Exxon Mobil	270772.0	25330.0
22001	1999	2	Ford Motor	144416.0	22071.0
24501	2004	2	Exxon Mobil	213199.0	21510.0
24507	2004	8	Citigroup	94713.0	17853.0
23000	2001	1	Exxon Mobil	210392.0	17720.0
25007	2005	8	Citigroup	108276.0	17046.0
25004	2005	5	General Electric	152363.0	16593.0
23501	2002	2	Exxon Mobil	191581.0	15320.0
24005	2003	6	Citigroup	100789.0	15276.0
24504	2004	5	General Electric	134187.0	15002.0
25017	2005	18	Bank of America Corp.	63324.0	14143.0
23506	2002	7	Citigroup	112022.0	14126.0
24004	2003	5	General Electric	131698.0	14118.0
23505	2002	6	General Electric	125913.0	13684.0
23005	2001	6	Citigroup	111826.0	13519.0
25005	2005	6	ChevronTexaco	147967.0	13328.0
23004	2001	5	General Electric	129853.0	12735.0
23009	2001	10	Verizon Communications	64707.0	11797.0
24002	2003	3	Exxon Mobil	182466.0	11460.0
25023	2005	24	Pfizer	52921.0	11361.0

Technically, the profit should be comparable, which means we need to compare the profit data after adjusting for inflation. However, due to my limited information about how this data was collected and preprocessed, I also provide the top 20 companies with highest profits within one year.

In [16]:

```
# Check for year

raw.groupby('Year').count()
```

Out[16]:

	Rank	Company	Revenue	Profit
Year				
1955	494	494	494	494
1956	498	498	498	498
1957	497	497	497	497
1958	497	497	497	497
1959	497	497	497	497
1960	499	499	499	499
1961	498	498	498	498
1962	498	498	498	498
1963	500	500	500	500
1964	500	500	500	500
1965	499	499	499	499
1966	500	500	500	500
1967	500	500	500	500
1968	500	500	500	500
1969	500	500	500	500
1970	500	500	500	500
1971	493	493	493	493
1972	492	492	492	492
1973	493	493	493	493
1974	493	493	493	493
1975	494	494	494	494
1976	493	493	493	493
1977	493	493	493	493
1978	491	491	491	491
1979	491	491	491	491
1980	493	493	493	493
1981	492	492	492	492
1982	490	490	490	490
1983	491	491	491	491
1984	489	489	489	489
1985	487	487	487	487
1986	477	477	477	477

	Rank	Company	Revenue	Profit
Year				
1987	477	477	477	477
1988	480	480	480	480
1989	476	476	476	476
1990	481	481	481	481
1991	480	480	480	480
1992	479	479	479	479
1993	484	484	484	484
1994	484	484	484	484
1995	494	494	494	494
1996	494	494	494	494
1997	496	496	496	496
1998	497	497	497	497
1999	497	497	497	497
2000	496	496	496	496
2001	497	497	497	497
2002	496	496	496	496
2003	495	495	495	495
2004	500	500	500	500
2005	499	499	499	499

In [17]:

```
# Print the top 20 companies with the highest profits in 2005

raw[raw['Year'] == 2005][:20]
```

Out[17]:

	Year	Rank	Company	Revenue	Profit
25001	2005	2	Exxon Mobil	270772.0	25330.0
25007	2005	8	Citigroup	108276.0	17046.0
25004	2005	5	General Electric	152363.0	16593.0
25017	2005	18	Bank of America Corp.	63324.0	14143.0
25005	2005	6	ChevronTexaco	147967.0	13328.0
25023	2005	24	Pfizer	52921.0	11361.0
25008	2005	9	American Intl. Group	98610.0	11050.0
25000	2005	1	Wal-Mart Stores	288189.0	10267.0
25016	2005	17	Altria Group	64440.0	9416.0
25029	2005	30	Johnson & Johnson	47348.0	8509.0
25009	2005	10	Intl. Business Machines	96293.0	8430.0
25040	2005	41	Microsoft	36835.0	8168.0
25006	2005	7	ConocoPhillips	121663.0	8129.0
25013	2005	14	Verizon Communications	71563.3	7830.7
25049	2005	50	Intel	34209.0	7516.0
25011	2005	12	Berkshire Hathaway	74382.0	7308.0
25051	2005	52	Wells Fargo	33876.0	7014.0
25025	2005	26	Procter & Gamble	51407.0	6481.0
25032	2005	33	SBC Communications	41098.0	5887.0
25083	2005	84	Merck	22938.6	5813.4

4. Using the data from data2.json, find out how many unique companies are in the data.

In [18]:

```
# Load JSON data as dataframe

data2 = pd.read_json (r'data2.json')
```

In [19]:

```
print('There are {} unique companies in the data.'.format(data2['Company'].
```

```
There are 1860 unique companies in the data.
```

```
In [20]: data2.nunique()
```

```
Out[20]: Year          51
         Rank         500
         Company     1860
         Revenue    18010
         Profit      6976
         dtype: int64
```

5. List top ten companies that have reported yearly data most often.

```
In [21]: # Find out the companies that reported yearly data most often

         data2['Company'].value_counts()[:73]
```

```
Out[21]: CBS                         57
         OfficeMax                   55
         Boeing                      51
         Avon Products               51
         McGraw-Hill                 51
                                     ..
         Abbott Laboratories         51
         United Technologies         51
         Navistar International      51
         Crown Holdings              51
         Lockheed Martin             50
         Name: Company, Length: 73, dtype: int64
```

The whole data set records 51 years of data from 1955 to 2005. There are 72 companies reporting all 51 years of data, including 2 companies reporting more than 51 years of data. For the two companies CBS and OfficeMax, they reported some of the yearly data twice. In those years reported twice, the other data like Rank, Revenue, Profit are also differ, so I will treat this as two subsidiaries of this parent company, but when calculating other data, I still regard it as one company. Therefore, it's hard to tell the top 10 companies that have reported data most often. I will give the name of all 72 companies as my result to this question.

In [22]:

```
# Print the name of all 72 companies reported their data in 51 years

data2['Company'].value_counts().index.tolist()[:72]
```

Out[22]:

```
['CBS',
 'OfficeMax',
 'Boeing',
 'Avon Products',
 'McGraw-Hill',
 'Archer Daniels Midland',
 'Bristol-Myers Squibb',
 'Colgate-Palmolive',
 'Pfizer',
 'Whirlpool',
 'Amerada Hess',
 'ConocoPhillips',
 'Anheuser-Busch',
 'Georgia-Pacific',
 'DuPont',
 'Motorola',
 'Marathon Oil',
 'Alcoa',
 'General Motors',
 'Sunoco',
 'Kellogg',
 'PPG Industries',
 'Honeywell Intl.',
 'Unisys',
 'Owens Corning',
 'Rockwell Automation',
 'Fortune Brands',
 'Exxon Mobil',
 'Goodyear Tire & Rubber',
 'Gillette',
 'Eastman Kodak',
 'USG',
 'Ashland',
 'Weyerhaeuser',
 'Kimberly-Clark',
 'Unocal',
 'Campbell Soup',
 'H.J. Heinz',
 'Merck',
 'General Electric',
 'Eli Lilly',
 'Intl. Business Machines',
 'Dana',
 'Cummins',
 'Procter & Gamble',
 'American Standard',
 'Textron',
 'Phelps Dodge',
 'Hormel Foods',
 'Caterpillar',
 'Eaton',
 'Paccar',
 '3M',
 'International Paper',
 'Rohm & Haas',
 'General Mills',
 'General Dynamics',
 'Deere',
 'Raytheon',
 'Northrop Grumman',
 'Wyeth',
 'Hershey Foods',
 'Coca-Cola',
 'ChevronTexaco',
 'Dow Chemical',
 'Altria Group',
 'Johnson & Johnson',
 'PepsiCo',
 'Abbott Laboratories',
 'United Technologies',
 'Navistar International',
 'Crown Holdings']
```

6. List the number of companies that have only reported data once.

In [23]:

```
# Check the companies that have only reported data once

cnt = data2['Company'].value_counts()
cnt[cnt == 1]
```

Out[23]:

```
Frederick & Herrud              1
New York Shipbuilding           1
Interim Services                1
Ivax                            1
Stater Bros. Holdings           1
                               ..
Pittsburgh Coke & Chemical      1
Assurant                        1
Wendy's International           1
Southern Industries             1
Continental                     1
Name: Company, Length: 182, dtype: int64
```

In [24]:

```
once = cnt[cnt==1]
print('There are {} companies that have only reported data once.'.format(le
```

```
There are 182 companies that have only reported data once.
```

In [25]:
```
# Print the name of all companies that have only reported data once

once.index.tolist()
```

Out[25]:
```
['Frederick & Herrud',
 'New York Shipbuilding',
 'Interim Services',
 'Ivax',
 'Stater Bros. Holdings',
 'Roadway',
 'Grand Union Holdings',
 'First Chicago Corp.',
 'MID-AMERICA DAIRYMEN',
 'Truax-Traer Coal',
 'SuCrest',
 'B.V.D.',
 'Visking',
 'Coca-Cola Bottling Co. of New York',
 'American Chicle',
 'Knoll International Holdings',
 'Cincinnati Financial',
 'NSTAR',
 'Kohler',
```

In [ ]:
```
```