Project 1: Baby Names

CS 5473: Data Mining Fall 2022

Instructor: Dr. Mohammad Imran Chowdhury

Total: 100 Points

Due: 9/15/2022 11:59 PM

In this project, I invite you to do the following tasks using the python libraries such as NumPy, Pandas, Matplotlib, etc. covered in class.

Task 1: Load the Dataset Baby Names provided to you as **``names.zip"** file into the Jupyter Notebook. Note that this notebook requires Python 3.6 or higher.

After the loading the dataset into the Jupyter Notebook, the output should look like as follows:

```
In [2]: import zipfile

In [3]: zipfile.ZipFile('names.zip').extractall('.')

In [4]: ls
 Volume in drive C is Windows
 Volume Serial Number is 80EA-62F4

 Directory of C:\Users\imran\CS 4373 Class Demo\Exercise Files\chapter7

08/03/2022  08:02 PM    <DIR>          .
08/03/2022  08:02 PM    <DIR>          ..
08/02/2022  06:45 PM    <DIR>          .ipynb_checkpoints
08/03/2022  07:59 PM             3,716 07_02_loading.ipynb
01/30/2020  10:00 PM             5,213 07_03_popularity.ipynb
01/30/2020  10:04 PM             5,425 07_04_topten.ipynb
08/03/2022  07:37 PM            13,384 07_06_solution.ipynb
08/03/2022  08:02 PM    <DIR>          names
01/29/2020  01:35 PM         8,528,645 names.zip
               5 File(s)      8,556,383 bytes
               4 Dir(s)  87,057,903,616 bytes free
```

Type ``ls names" to list all files under the unzipped names folder. The output should be as follows: **(5 points)**

```
In [5]: ls names
 Volume in drive C is Windows
 Volume Serial Number is 80EA-62F4

 Directory of C:\Users\imran\CS 4373 Class Demo\Exercise Files\chapter7\names

08/03/2022  08:02 PM    <DIR>          .
08/03/2022  08:02 PM    <DIR>          ..
08/03/2022  08:02 PM           316,364 NationalReadMe.pdf
08/03/2022  08:02 PM            24,933 yob1880.txt
08/03/2022  08:02 PM            24,065 yob1881.txt
08/03/2022  08:02 PM            26,559 yob1882.txt
08/03/2022  08:02 PM            26,002 yob1883.txt
08/03/2022  08:02 PM            28,670 yob1884.txt
08/03/2022  08:02 PM            28,625 yob1885.txt
08/03/2022  08:02 PM            29,822 yob1886.txt
08/03/2022  08:02 PM            29,531 yob1887.txt
08/03/2022  08:02 PM            33,064 yob1888.txt
08/03/2022  08:02 PM            32,297 yob1889.txt
08/03/2022  08:02 PM            33,621 yob1890.txt
```

Note: All these .txt files contain comma-separated values (CSV).

Task 2: Load CSV file 'names/yob2011.txt' as DataFrame, then create a new column "year" with all elements set to 2011. The output should be as follows: **(10 points)**

Out[9]:

	name	sex	number	year
0	Sophia	F	21842	2011
1	Isabella	F	19910	2011
2	Emma	F	18803	2011
3	Olivia	F	17322	2011
4	Ava	F	15503	2011
...	...	...	...	...
33903	Zylar	M	5	2011
33904	Zylas	M	5	2011
33905	Zyran	M	5	2011
33906	Zyshawn	M	5	2011
33907	Zytavion	M	5	2011

33908 rows × 4 columns

Task 3: For each year in 1880-2018, load the corresponding CSV file 'names/yobXXXX.txt' as DataFrame, create new column "year" with all elements set to loop variable, then concatenate all DataFrames into a single one named as **``allyears"**. After issuing the ``allyears.info()" command the output should be as follows: **(20 points)**

```
In [11]: allyears.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 1957046 entries, 0 to 32032
Data columns (total 4 columns):
name      object
sex       object
number    int64
year      int64
dtypes: int64(2), object(2)
memory usage: 74.7+ MB
```

Save the DataFrame as ``allyears.csv.gz'' to compressed CSV file, dropping uninteresting index. This should be done as follows: **(5 points)**

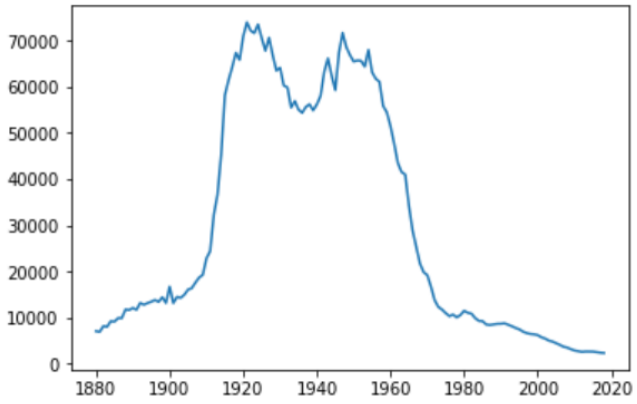
```
allyears.to_csv('allyears.csv.gz', index=False)
```

Task 4: First load and read the combine DateFrame **``allyears.csv.gz''**. Then, we want examine the changing popularity of a name. So, we need to reframe the DataFrame as

``allyears_indexed"** to make things easier. To do that use multi-index: set ``sex" first, then ``name" and then ``year". After that sort the index by calling ``sort_index()" method. **(10 points)

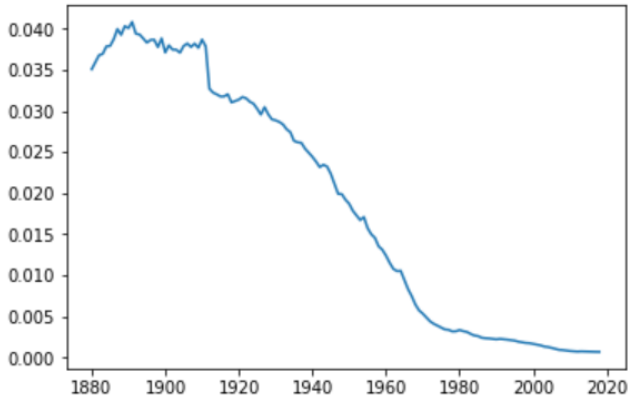
Now get the Data for a given ``name" -> "Mary" and ``sex" -> ``F" and then plot it. You can use the loc method to get the data say .loc[('F', 'Mary')]. After plotting the data, the output should be as follows: **(10 points)**

```
Out[6]: [<matplotlib.lines.Line2D at 0x26596f8a4a8>]
```



Next, normalize the above ``F/Mary time series plot" by the total number of births each year. The output should be as follows: **(10 points)**

```
Out[7]: [<matplotlib.lines.Line2D at 0x265971d2c50>]
```



Hints: To normalize, you should use .groupby('year').sum() method.

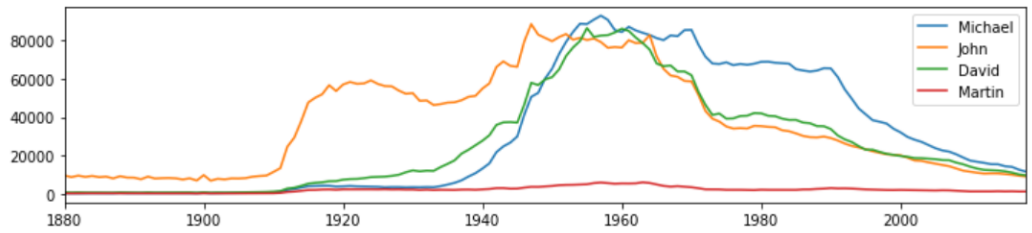
Tasks 5: You need to write two methods.

First method should be named as ``plotname(sex, name)" to plot number of sex/name babies as a function of year **(15 points)**.

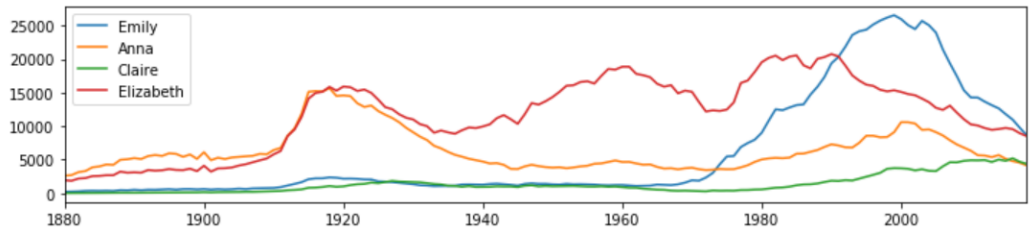
The second method should be named as ``comparenames(sex, names)" to combine several ``plotname()" plots for given sex and list of names. **(15 points)**

The following shows the plotting results of two sample runs of the ``comparenames(sex, names)" method. Your program output should be same as mine:

```
In [10]: comparenames('M', ['Michael','John','David','Martin'])
```




```
In [11]: comparenames('F', ['Emily','Anna','Claire','Elizabeth'])
```



The submission grading rubric is as follows (points out of 100 total):

Project element	**Points**
Task 1	5
Task 2	10
Task 3	25
Task 4	30
Task 5	30

Submission Instructions: Create a compressed file (.zip or .tar.gz files are accepted) with your all source files such as .ipynb files and data files. Generally speaking to complete Task1 through Task5, you just need one .ipynb file. But it's better to submit everything as a compressed file. Submit the compressed file to Blackboard.

Late submission policy: As described in the syllabus, any late submission will the penalized with 10% off after each 24 hours late. For example, an assignment worth 100 points turned in 2 days late will receive a 20 point penalty. Assignments turned in 5 or more days after the due date will receive a grade of 0.