PA 1: Exploratory Analysis using Given Datasets

Student Details

Student Name and ID:

Notes: When submitting, fill your name and ID in this cell. Note that this is a markdown cell! Do not make any changes in the dataset file and do not rename the 'database.csv'. Rename your submission file to **'yourLastName_Last4digitsofyourID_PA1.ipynb'**. (This name is long, have to have 3 member's name). Do not to forget to cite any external sources used by you. [2.5 points]

Assignment Details

In this assignment, you will conduct a guided exploration over the given dataset.

You will prepare a report with the following outline for each one of the datasets. Look at the following Example.

1. Introduction
2. Retrieving the Data
3. Glimpse of Data
4. Check for missing data
5. Data Exploration

You will learn and use some of the most common **exploration/aggregation/descriptive** operations. This should also help you learn most of the key functionalities in Python/Pandas, Weka and R.

You will also learn how to use visualization libraries to identify patterns in data that will help in your further data analysis. You will also explore most popular chart types and how to use different libraries and styles to make your visualizations more attractive.

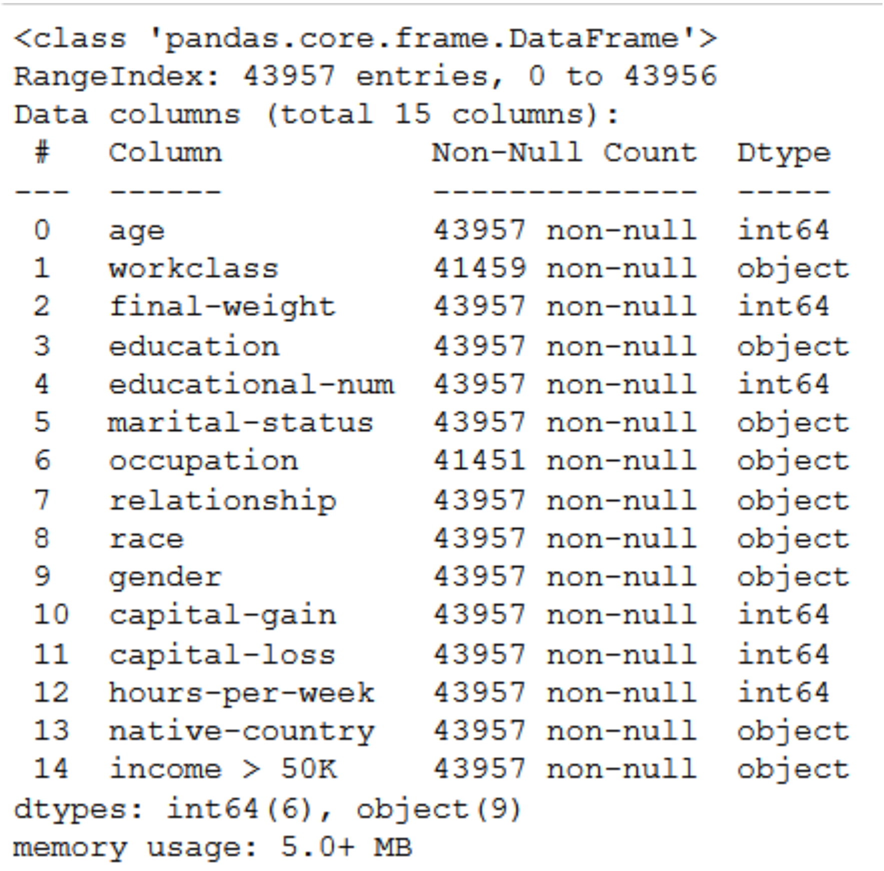
DO Task 1, Task 2, Task 3, Task 4 using Python/Pandas, Weka, R

Out of the 3 datasets listed below:
1. Income dataset should be solved using Python in Jupyter notebook only.
2. Surgical dataset using WEKA
3. Healthcare_stroke_dataset using R

Dataset Details

In this assignment, you will work on 1)

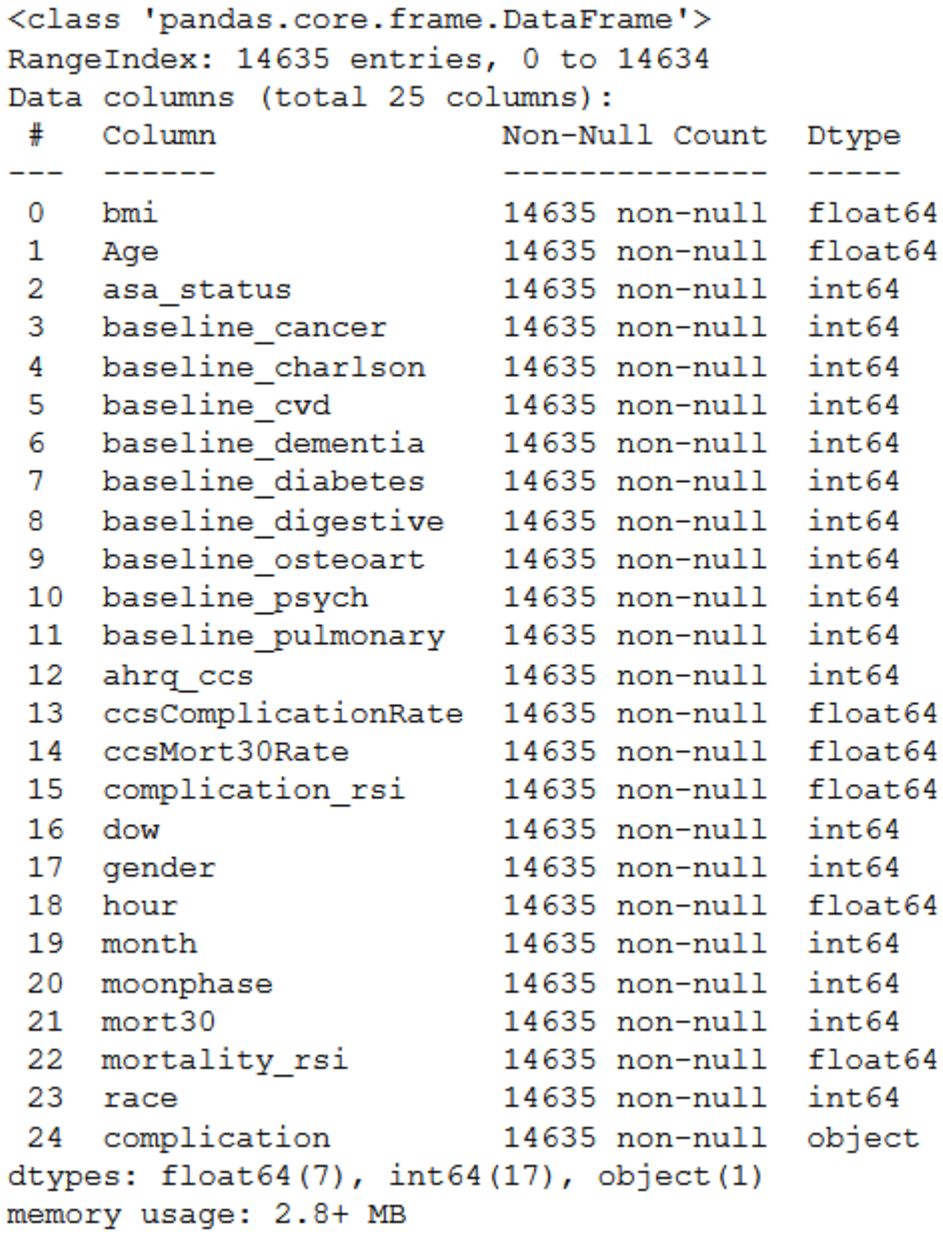
Income_dataset contains 43957 rows and 15 columns. The columns of the data-set are:



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 43957 entries, 0 to 43956
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   age              43957 non-null  int64 
 1   workclass        41459 non-null  object
 2   final-weight     43957 non-null  int64 
 3   education        43957 non-null  object
 4   educational-num  43957 non-null  int64 
 5   marital-status   43957 non-null  object
 6   occupation       41451 non-null  object
 7   relationship     43957 non-null  object
 8   race             43957 non-null  object
 9   gender           43957 non-null  object
 10  capital-gain     43957 non-null  int64 
 11  capital-loss     43957 non-null  int64 
 12  hours-per-week   43957 non-null  int64 
 13  native-country   43957 non-null  object
 14  income > 50K     43957 non-null  object
dtypes: int64(6), object(9)
memory usage: 5.0+ MB
```

2) Another dataset Surgical-deepnet

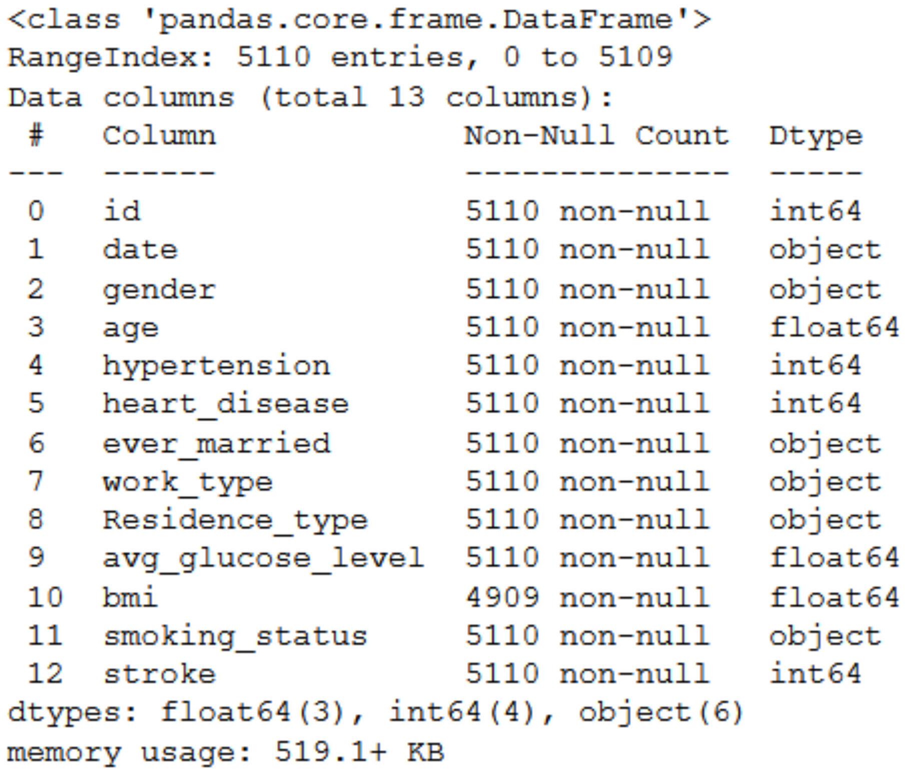
Content: Dataset contains 14635 rows and 25 columns



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14635 entries, 0 to 14634
Data columns (total 25 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   bmi                  14635 non-null  float64
 1   Age                  14635 non-null  float64
 2   asa_status           14635 non-null  int64  
 3   baseline_cancer      14635 non-null  int64  
 4   baseline_charlson    14635 non-null  int64  
 5   baseline_cvd         14635 non-null  int64  
 6   baseline_dementia    14635 non-null  int64  
 7   baseline_diabetes    14635 non-null  int64  
 8   baseline_digestive   14635 non-null  int64  
 9   baseline_osteoart    14635 non-null  int64  
 10  baseline_psych       14635 non-null  int64  
 11  baseline_pulmonary   14635 non-null  int64  
 12  ahrq_ccs             14635 non-null  int64  
 13  ccsComplicationRate  14635 non-null  float64
 14  ccsMort30Rate        14635 non-null  float64
 15  complication_rsi     14635 non-null  float64
 16  dow                  14635 non-null  int64  
 17  gender               14635 non-null  int64  
 18  hour                 14635 non-null  float64
 19  month                14635 non-null  int64  
 20  moonphase            14635 non-null  int64  
 21  mort30               14635 non-null  int64  
 22  mortality_rsi        14635 non-null  float64
 23  race                 14635 non-null  int64  
 24  complication         14635 non-null  object 
dtypes: float64(7), int64(17), object(1)
memory usage: 2.8+ MB
```

3) Healthcare_stroke_dataset.csv

Content: Dataset contains 5110 rows and 13 columns



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5110 entries, 0 to 5109
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   id                 5110 non-null   int64  
 1   date               5110 non-null   object 
 2   gender             5110 non-null   object 
 3   age                5110 non-null   float64
 4   hypertension       5110 non-null   int64  
 5   heart_disease      5110 non-null   int64  
 6   ever_married       5110 non-null   object 
 7   work_type          5110 non-null   object 
 8   Residence_type     5110 non-null   object 
 9   avg_glucose_level  5110 non-null   float64
 10  bmi                4909 non-null   float64
 11  smoking_status     5110 non-null   object 
 12  stroke             5110 non-null   int64  
dtypes: float64(3), int64(4), object(6)
memory usage: 519.1+ KB
```

Required Python Packages

You will use the packages imported below in this assignment. Do NOT import any new packages without confirming with the TA.

In [1]:

```
# special IPython command to prepare the notebook for matplotlib
%matplotlib inline

#Array processing
import numpy as np
#Data analysis, wrangling and common exploratory operations
import pandas as pd
from pandas import Series, DataFrame

#For visualization. Matplotlib for basic viz and seaborn for more stylish figu
res
import matplotlib.pyplot as plt
import seaborn as sns
```

Reading Dataset

The Python code below reads the Income dataset dataset into a Pandas data frame with the name df_data. For this code to work, the file ' Income_dataset.csv' must be in the same folder as this file.

```
#read the csv file into a Pandas data frame
df_data = pd.read_csv('income_dataset.csv', encoding='latin1')

#return the first 5 rows of the dataset
df_data.head()
```

	age	workclass	final-weight	education	educational-num	marital-status	occupation	relationship	race	gender	capital-gain	capital-loss	hours-per-week	native-country	income > 50K
0	67	Private	366425	Doctorate	16	Divorced	Exec-managerial	Not-in-family	White	Male	99999	0	60	United-States	Yes
1	17	Private	244602	12th	8	Never-married	Other-service	Own-child	White	Male	0	0	15	United-States	No
2	31	Private	174201	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	40	United-States	Yes
3	58	State-gov	110199	7th-8th	4	Married-civ-spouse	Transport-moving	Husband	White	Male	0	0	40	United-States	No
4	25	State-gov	149248	Some-college	10	Never-married	Other-service	Not-in-family	Black	Male	0	0	40	United-States	No

Task 1: Statistical Exploratory Data Analysis

Let us start with getting to know the dataset. Your first task will be to get some basic information by using Pandas features. Do task 1 for each Dataset.

```
#For each task below, look for a Pandas function to do the task. #Replace
None in each task with your code.

# 2.5 points
#Task 1-a: Print the details of the df_data data frame (information such as
nu mber of rows,columns, name of columns, etc)
print (">>Task 1-a: Details of df_data data frame are: \n", None )

# 2.5 points
#Task 1-b: Find the number of rows and columns in the df_data data frame.
num_rows = None
num_cols = None
print ("\n\n>>Task 1-b: Number of rows:%s and number of columns:%s" %
(num_row s, num_cols))

# 2.5 points
#Task 1-c: Print the descriptive detail (count, unique, top, freq etc) for
'educational-num'' column of the df_data

print ("\n\n>>Task 1-c: Descriptive details of ‘educational-num' column
are\n",None
)
```

```
    # 10 points
    #Task 1-d: Print ALL the unique values of Capital-gain.


    # create new dataframe, repeating or chaining as appropriate              .
    num_uniq_capital_gain= None
    num_uniq_county = None

    print ("\n\n >>Task 1-d:") print(num_uniq_capital_gain)
    print("#####################################################")
    print(num_uniq_county)
```

In [3]:

```
>>Task 1-a: Details of df_data data frame are:
None

>>Task 1-b: Number of rows: None and number of columns:None

>>Task 1-c: Print the descriptive detail (count, unique, top, freq etc) for 'educational-
num'' column of the df_data
None

>>Task 1-d: Print ALL the unique values of Capital-gain
None #####################################################
None
```

Task 2: Aggregation & Filtering & Rank

In this task, we will perform some very high-level aggregation and filtering operations. Then, we will apply ranking on the results for some tasks. Pandas has a convenient and powerful syntax for aggregation, filtering, and ranking. DO NOT write a for loop. Pandas has built-in functions for all tasks.

```
In [4]:      # 8 points
             #Task 2-a: Find out the race with largest number of records

             Race_greater = None
             print (">>Task 2-a: The Race with the largest number of records is %s"
             % (Race_greater))

             # 8 points
             #Task 2-b: Find out the total number of doctorate who are married
             #

             num_doctorate = None

            pprint ("\n\n>>Task 2-b: The total number of doctorate who are married %s"
             % (num_doctorate))

             # 14 points
             #Task 2-c: Find out the top 10 countries with the highest income.
             n = 10
             top10_countries=None
             top10_male=None
            print ("\n\n>>Task 2-c: top 10 countries with the highest income: \n%s" %
             (top10_countries))
             print ("\n\n>>Task 2-c: top 10 counties with the most male \n%s" % (top10_male))
```

Task 3: Visualization (30 points)

In this task, you will perform a number of visualization tasks to get some intuition about the data. Visualization is a key component of exploration. You can choose to use either Matplotlib or Seaborn for plotting. The default figures generated from Matplotlib may look unaesthetic and so you might want to try Seaborn to get better figures. Seaborn has a variety of styles. Feel free to experiment with them and choose the one you like. We have assigned 10 points for the aesthetics of your visualizations.

```
sns.set.stye(‘whitegrid’)
sns.set(font_scale = 1.3)
# 10 points
# Task 3-a: Plot the race count for each country
# think of a nice way to visualize all the countries.
# 20 points
# task 3b: Draw a pie chart that represents native country
```

Task 4:

Find out an 'interesting' information from each one of the dataset. Create a visualization for it and explain in a few lines your reasoning.

This task is worth 20 points. Your result will be judged based on the uniqueness and quality of your work (having a meaningful result and an aesthetic visualization).

In [6]:

```
#########################begin code for Task 4

#########################end code for Task 4
```