PYTHON: HIGH PERFORMANCE PANDAS

INTRODUCTION

This assignment will cover high performance pandas techniques focusing on optimizing memory usage.

DATA

For this assignment, we will be using some simulated electronic health records (these are not real data!). This is a common sort of dataset for health care systems to use when tracking all the patients and the outpatient activity. You should take a few minutes to review the datasets using Excel, read the descriptions, and understand how they fit together. We will only use a few datasets in this exercise, but you could explore all of them on your own.

- OutpatientVisit.csv
  - This is a table with a row for every outpatient visit. This file can be linked to the patient table using Patient ID, the clinic table using ClinicID, the ICD table using the ICD_1-3 columns, and the staff table using StaffID.
- Patient.csv
  - This file has one row for every patient, and includes demographic information about each patient. This file can be linked to the outpatient visit file using Patient ID.
- Clinic.csv
  - This is a table of all the clinic IDs, and if the clinic is primary care, specialty care, or emergency department. There should be one row per clinic ID. You can link this file to the OutpatientVisit file by clinic code to see if a visit occurred in a primary care clinic, a specialty clinic, or the emergency department.
- Staff.csv
  - This is a table with information about each staff member. This can be linked to the visit file using staffID
- Mortality.csv
  - This is a table of date of death for each patient (if they have died). Patients who have not died are not listed in this file. You can link this to the other files using PatientID

- ICDCodes.csv
    - This is a table with the description for each ICD code. ICD stands for 'international classification of disease'. Each outpatient visit is associated with 1-3 of these codes, indicating what diseases the patient has that were addressed during the visit.
- DiseaseMap.csv

This is a table that maps different ICD codes to clinically meaningful categories. For example, there are many different diagnoses for diabetes, but we might wish to generally call all of them 'diabetes'.

QUESTIONS

1) Manual data compression using Pandas dtypes: In this first question, you will manually decrease the size of a dataset by examining the first 100 rows of the dataset, then deciding what dtypes to use for the full dataframe import.
    a. Import only the first 100 rows of the patient.csv dataset using Python/pandas.
    b. Examine the column names and the dtypes of the dataframe
    c. Create a dict of columns names and types using the to_dict() method
    d. Decide which columns you can compress by specifying a smaller dtype. For example, the default dtype of an integer is int64, but you may be able to fit hat integer data into the dtype int32, or int16, or uint16 (unsigned integer). It depends on the data! Consider turning the text data into categorical data. Try to make the dataframe as small as reasonably possible.
    e. Use the memory_usage(deep=True) dataframe method to calculate large your reduced file is.
    f. Import the patient.csv dataframe with default datatypes and calculate the memory_usage(deep=True). How much smaller is your reduced dataframe than the full dataframe?
    g. Repeat a-f for the OutpatientVisit.csv file.

    Here is some sample code using the gapminder data to help you with this question, and some info on dtypes after that!


```
import pandas as pd
# import first 10 rows of data to explore dtypes
dt1 = pd.read_csv('data/gapminder.csv',nrows=10)
dt1.columns
dt1.dtypes
# manual downcast of existing dataframe to explore result
pd.to_numeric(dt1['year'],downcast='signed')
pd.to_numeric(dt1['year'],downcast='unsigned')

# now we will do it on import
# we need to create a dict of {column:data types}
# here I capture the dtypes as a dict, then just modify the one of interest afterwards
col_types = dt1.dtypes.to_dict()
# modify year to unsigned integer 16 bit
col_types['year'] = 'uint16'
# modify continent to be categorical
col_types['continent'] = 'category'

# Now apply the dtypes on import, import full file
dt2 = pd.read_csv('data/gapminder.csv',
                  dtype=col_types)
# Now year is an unsigned integer of length 16
dt2.columns
dt2.dtypes

## import the full file with default dtypes
dt3 = pd.read_csv('data/gapminder.csv')

## compare memory usage
dt3.info(memory_usage='deep')
dt3.memory_usage(deep=True)
dt3.memory_usage(deep=True).sum()
dt2.info(memory_usage='deep')
dt2.memory_usage(deep=True)
dt2.memory_usage(deep=True).sum()

## difference in memory
mem_diff = \
   dt2.memory_usage(deep=True).sum() - dt3.memory_usage(deep=True).sum()
## percent change
mem_diff / dt2.memory_usage(deep=True).sum()
```

Python Data Types		
Description	**Pandas Default**	**Available dtypes**
Strings	'object'	'object', 'category'
Continuous numbers (floats)	'float64'	'float16', 'float32', 'float64'
Integers	'int64'	'int8', 'int16', 'int32', 'int64', 'uint8', 'uint16', 'uint32', 'uint64'
Boolean	'bool'	'bool'
DateTime values	'datetime64'	'datetime64'