HDS 5230 High Performance Computing - HW5

February 20, 2019

Author: Miao Cai

```
In [ ]: import pandas as pd
        import os
        print("My working directory:\n" + os.getcwd())
        # os.chdir(r"C:\Users\evancarey\Dropbox\Work\SLU\Courses")
```

a. import only the first 100 rows of the patient.csv dataset using Python/pandas

```
In [22]: pt = pd.read_csv('healthcare2/Patient.csv', nrows = 100)
         pt.loc[0:10,:]
```

```
Out[22]:     PatientID  FirstName    LastName State  ZipCode DateOfBirth  Gender  \
         0           1      Diana  Huddleston    WI    53186  1962-02-27  female
         1           2     Marion      Poston    IL    60527  1859-09-11    male
         2           3     Sandra       Hamby    IL    60126  1946-02-15  female
         3           4    Mildred    Krehbiel    ID    83702  1979-07-27  female
         4           5    Abigail      Flores    PA    19131  1983-02-19  female
         5           6      Rusty      Thomas    AL    36107         NaN    male
         6           7     Robert   Alexander    CA    94539  1958-01-11    male
         7           8     Krista        Ward    WI    53219  1952-10-31  female
         8           9      Marti   Calabrese    MS    38801  1951-10-06  female
         9          10     Jeremy         Liu    CA    95526  1954-10-16    male
         10         11  Catherine       Tatum    MI    48213  1983-07-12  female

                Race       Income
         0       NaN  1076.167979
         1     white   475.781094
         2     white    30.747987
         3     white   160.596425
         4         ?          NaN
         5     black   171.378008
         6   Missing    66.226314
         7     black    15.078950
         8   Missing   114.598911
         9     white  1081.877157
         10  Missing    35.058641
```

b. Examine the column names and the dtypes of the dataframe

```
In [23]: pt.columns
```

```
Out[23]: Index(['PatientID', 'FirstName', 'LastName', 'State', 'ZipCode', 'DateOfBirth',
                'Gender', 'Race', 'Income'],
               dtype='object')
```

```
In [24]: pt.dtypes
```

```
Out[24]: PatientID        int64
         FirstName       object
         LastName        object
         State           object
         ZipCode          int64
         DateOfBirth     object
         Gender          object
         Race            object
         Income         float64
         dtype: object
```

c. Create a dict of columns names and types using the to_dict() method

```
In [25]: col_types = pt.dtypes.to_dict()
         col_types
```

```
Out[25]: {'PatientID': dtype('int64'),
          'FirstName': dtype('O'),
          'LastName': dtype('O'),
          'State': dtype('O'),
          'ZipCode': dtype('int64'),
          'DateOfBirth': dtype('O'),
          'Gender': dtype('O'),
          'Race': dtype('O'),
          'Income': dtype('float64')}
```

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.

```
In [26]: pt.info(memory_usage='deep')
         col_types['PatientID']='uint16'
         col_types['FirstName']='category'
         col_types['LastName']='category'
         col_types['State']='category'
         col_types['ZipCode']='uint16'
         col_types['DateOfBirth']='category'
         col_types['Gender']='category'
         col_types['Race']='category'
         col_types['Income']='float32'
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 9 columns):
PatientID      100 non-null int64
FirstName      100 non-null object
LastName       100 non-null object
State          100 non-null object
ZipCode        100 non-null int64
DateOfBirth    92 non-null object
Gender         98 non-null object
Race           97 non-null object
Income         92 non-null float64
dtypes: float64(1), int64(2), object(6)
memory usage: 38.8 KB
```

e. Use the memory_usage(deep=True) dataframe method to calculate large your reduced file is.

```
In [27]: pt_reduced = pd.read_csv('healthcare2/Patient.csv', nrows = 100,
                             dtype=col_types)
         pt_reduced.info(memory_usage='deep')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 9 columns):
PatientID      100 non-null uint16
FirstName      100 non-null category
LastName       100 non-null category
State          100 non-null category
ZipCode        100 non-null uint16
DateOfBirth    92 non-null category
Gender         98 non-null category
Race           97 non-null category
Income         92 non-null float32
dtypes: category(6), float32(1), uint16(2)
memory usage: 30.7 KB
```

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?

```
In [28]: pt1 = pd.read_csv('healthcare2/Patient.csv')
         pt1.info(memory_usage='deep')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 9 columns):
```

```
PatientID      20000 non-null int64
FirstName      20000 non-null object
LastName       20000 non-null object
State          20000 non-null object
ZipCode        20000 non-null int64
DateOfBirth    19000 non-null object
Gender         19431 non-null object
Race           19144 non-null object
Income         18600 non-null float64
dtypes: float64(1), int64(2), object(6)
memory usage: 7.6 MB
```

```
In [29]: pt1_reduced = pd.read_csv('healthcare2/Patient.csv',dtype=col_types)
         pt1_reduced.info(memory_usage='deep')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 9 columns):
PatientID      20000 non-null uint16
FirstName      20000 non-null category
LastName       20000 non-null category
State          20000 non-null category
ZipCode        20000 non-null uint16
DateOfBirth    19000 non-null category
Gender         19431 non-null category
Race           19144 non-null category
Income         18600 non-null float32
dtypes: category(6), float32(1), uint16(2)
memory usage: 2.8 MB
```

g. Repeat a-f for the OutpatientVisit.csv file.

```
In [30]: op0 = pd.read_csv('healthcare2/OutpatientVisit.csv', nrows = 100)
         op0.loc[0:10,:]
```

```
Out[30]:     VisitID  StaffID  PatientID   VisitDate ICD10_1 ICD10_2  ICD10_3  \
         0         1       46          1  2013-08-10  E10621    K269      NaN
         1         2       50          1  2013-12-02    K269  E10621      NaN
         2         3       13          1  2014-06-29  E10621    K269      NaN
         3         4       23          1  2014-09-19    K269  E10621      NaN
         4         5        9          1  2015-05-29    K269  E10621      NaN
         5         6       46          1  2016-05-07  E10621    K269      NaN
         6         7        7          1  2016-10-07  E10621    K269      NaN
         7         8       18          1  2016-11-07    K269  E10621      NaN
         8         9       23          1  2017-01-14    K269  E10621      NaN
         9        10        5          1  2017-01-29  E10621    K269      NaN
         10       11        2          1  2017-06-29    K269  E10621      NaN
```

```
    ClinicCode
0           15
1           55
2            1
3            3
4            5
5           15
6           41
7           31
8            3
9           14
10          55
```

```
In [31]: op0.columns
```

```
Out[31]: Index(['VisitID', 'StaffID', 'PatientID', 'VisitDate', 'ICD10_1', 'ICD10_2',
                'ICD10_3', 'ClinicCode'],
               dtype='object')
```

```
In [32]: op0.dtypes
```

```
Out[32]: VisitID         int64
         StaffID         int64
         PatientID       int64
         VisitDate      object
         ICD10_1        object
         ICD10_2        object
         ICD10_3       float64
         ClinicCode      int64
         dtype: object
```

```
In [33]: col_types = op0.dtypes.to_dict()
         col_types
```

```
Out[33]: {'VisitID': dtype('int64'),
          'StaffID': dtype('int64'),
          'PatientID': dtype('int64'),
          'VisitDate': dtype('O'),
          'ICD10_1': dtype('O'),
          'ICD10_2': dtype('O'),
          'ICD10_3': dtype('float64'),
          'ClinicCode': dtype('int64')}
```

```
In [34]: pt.info(memory_usage='deep')
         col_types['VisitID']='uint16'
         col_types['StaffID']='uint16'
         col_types['PatientID']='uint16'
         col_types['VisitDate']='category'
         col_types['ICD10_1']='category'
         col_types['ICD10_2']='category'
         col_types['ICD10_3']='category'
         col_types['ClinicCode']='uint16'
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 9 columns):
PatientID      100 non-null int64
FirstName      100 non-null object
LastName       100 non-null object
State          100 non-null object
ZipCode        100 non-null int64
DateOfBirth    92 non-null object
Gender         98 non-null object
Race           97 non-null object
Income         92 non-null float64
dtypes: float64(1), int64(2), object(6)
memory usage: 38.8 KB
```

```
In [35]: op0_reduced = pd.read_csv('healthcare2/OutpatientVisit.csv', nrows = 100,
                             dtype=col_types)
         op0_reduced.info(memory_usage='deep')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 8 columns):
VisitID       100 non-null uint16
StaffID       100 non-null uint16
PatientID     100 non-null uint16
VisitDate     100 non-null category
ICD10_1       100 non-null category
ICD10_2       63 non-null category
ICD10_3       0 non-null category
ClinicCode    100 non-null uint16
dtypes: category(4), uint16(4)
memory usage: 12.7 KB
```

```
In [36]: op = pd.read_csv('healthcare2/OutpatientVisit.csv')
         op.info(memory_usage='deep')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 174690 entries, 0 to 174689
Data columns (total 8 columns):
VisitID       174690 non-null int64
StaffID       174690 non-null int64
PatientID     174690 non-null int64
```

```
VisitDate     173252 non-null object
ICD10_1       174690 non-null object
ICD10_2       59785 non-null object
ICD10_3       19362 non-null object
ClinicCode    174690 non-null int64
dtypes: int64(4), object(4)
memory usage: 39.7 MB
```

```
In [37]: op_reduced = pd.read_csv('healthcare2/OutpatientVisit.csv',dtype=col_types)
         op_reduced.info(memory_usage='deep')
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 174690 entries, 0 to 174689
Data columns (total 8 columns):
VisitID       174690 non-null uint16
StaffID       174690 non-null uint16
PatientID     174690 non-null uint16
VisitDate     173252 non-null category
ICD10_1       174690 non-null category
ICD10_2       59785 non-null category
ICD10_3       19362 non-null category
ClinicCode    174690 non-null uint16
dtypes: category(4), uint16(4)
memory usage: 3.7 MB
```