labassignment9

July 30, 2022

1 Lab Assignment 9: Data Management Using pandas, Part 2

1.1 DS 6001: Practice and Application of Data Science

1.1.1 Instructions

Please answer the following questions as completely as possible using text, code, and the results of code as needed. Format your answers in a Jupyter notebook. To receive full credit, make sure you address every part of the problem, and make sure your document is formatted in a clean and professional way.

1.2 Problem 0

Import the following libraries:

[1]:
```
import numpy as np
import pandas as pd
```

1.3 Problem 1

In the first part of this lab, the goal is to merge data from the United Nations World Health Organization (https://www.who.int/who-un/en/) with data from the Varieties of Democracy Project (https://www.v-dem.net/en/). The UN-WHO studies health outcomes in a cross-national context, and V-Dem studies the quality of democracy as it changes across countries and over time. We would want to merge these two datasets together if we wanted to study whether democratic quality can predict health outcomes.

The UN data contains cross-national time series data from the United Nations and World Health Organization, and includes three features:

- The number of physicians per 1000 people
- The percent of the population that is malnourished
- Health expenditure per capita

The VDem data comes from the Varieties of Democracy project, which aims to measure the quality of democracy and the amount of corruption in different countries over time (https://www.v-dem.net/en/data/data-version-8/). This data file contains indices regarding a country's democractic quality, level of civil liberites, and corruption. It also contains a binary indicator that separates countries into democratic and nondemocratic states, and it includes a categorizaton of the corruption scale.

The URLs for the two datasets are:

[2]:
```
undata_url = "https://github.com/jkropko/DS-6001/raw/master/localdata/UNdata.
 ↪csv"
VDem_url = "https://github.com/jkropko/DS-6001/raw/master/localdata/vdem.csv"
```

1.3.1 Part a

Load both CSV files. Make sure to check whether there are rows that should not be included in the dataframe, and whether there are missing codes that should be replaced with `NaN`. Fix these problems at the data loading stage, if you can. (Don't worry about column names or category labels yet.) Also, the UN data covers the years 1960-2014, and the VDem data covers the years 1960-2015. To make the timeframe match up, delete rows in the VDem data from 2015. (1 point)

[3]:
```
UN = pd.read_csv(undata_url,skipfooter=2,engine='python')
UN = UN.dropna(how='all')
UN = UN.replace('..',np.nan)
UN
```

[3]:
```
                                    Series Name     Series Code  \
0                 Physicians (per 1,000 people)  SH.MED.PHYS.ZS
1                 Physicians (per 1,000 people)  SH.MED.PHYS.ZS
2                 Physicians (per 1,000 people)  SH.MED.PHYS.ZS
3                 Physicians (per 1,000 people)  SH.MED.PHYS.ZS
4                 Physicians (per 1,000 people)  SH.MED.PHYS.ZS
..                                          …              …
769  Health expenditure per capita (current US$)     SH.XPD.PCAP
770  Health expenditure per capita (current US$)     SH.XPD.PCAP
771  Health expenditure per capita (current US$)     SH.XPD.PCAP
772  Health expenditure per capita (current US$)     SH.XPD.PCAP
773  Health expenditure per capita (current US$)     SH.XPD.PCAP

           Country Name Country Code       1960 [YR1960] 1961 [YR1961]  \
0           Afghanistan          AFG  0.0348442494869232           NaN
1               Albania          ALB   0.276291221380234           NaN
2               Algeria          DZA   0.173148155212402           NaN
3        American Samoa          ASM                 NaN           NaN
4               Andorra          ADO                 NaN           NaN
..                  …            …                   …             …
769  West Bank and Gaza          WBG                 NaN           NaN
770               World          WLD                 NaN           NaN
771         Yemen, Rep.          YEM                 NaN           NaN
772              Zambia          ZMB                 NaN           NaN
773            Zimbabwe          ZWE                 NaN           NaN

    1962 [YR1962] 1963 [YR1963] 1964 [YR1964]       1965 [YR1965]  …  \
0             NaN           NaN           NaN  0.0634277984499931  …
1             NaN           NaN           NaN    0.48128342628479  …
2             NaN           NaN           NaN   0.116413652896881  …
```

```
3             NaN           NaN           NaN                 NaN  …
4             NaN           NaN           NaN                 NaN  …
..            …             …             …                   …  …
769           NaN           NaN           NaN                 NaN  …
770           NaN           NaN           NaN                 NaN  …
771           NaN           NaN           NaN                 NaN  …
772           NaN           NaN           NaN                 NaN  …
773           NaN           NaN           NaN                 NaN  …

       2006 [YR2006]     2007 [YR2007]     2008 [YR2008]     2009 [YR2009]  \
0              0.136             0.146             0.145             0.175
1               1.15             1.146               NaN             1.144
2                NaN             1.207               NaN               NaN
3                NaN               NaN               NaN               NaN
4               3.64             3.716               NaN             3.912
..               …                 …                 …                 …
769              NaN               NaN               NaN               NaN
770  748.814060401144  822.305554616346  895.873122675708  906.913628537332
771       52.0957528       58.08815663        69.6851483       65.91392692
772       62.92278027        48.1756618       66.53854373       53.65967398
773       21.24877429       17.80401666       16.21371054       37.20702422

       2010 [YR2010]     2011 [YR2011]     2012 [YR2012]     2013 [YR2013]  \
0              0.194             0.234             0.225             0.266
1              1.132             1.113             1.145             1.145
2              1.207               NaN               NaN               NaN
3                NaN               NaN               NaN               NaN
4                  4               NaN               NaN               NaN
..               …                 …                 …                 …
769              NaN               NaN               NaN               NaN
770  948.697373861547  1019.76342698985  1026.15480217145  1041.74995700284
771       67.74874258       64.65160379       73.90669273       78.52249036
772       64.17510388       70.52581827       82.86919808       87.83302346
773       36.36279411       48.46958014       57.25376348       62.30922835

       2014 [YR2014] 2015 [YR2015]
0                NaN           NaN
1                NaN           NaN
2                NaN           NaN
3                NaN           NaN
4                NaN           NaN
..               …             …
769              NaN           NaN
770  1060.98712764105           NaN
771       79.93696624           NaN
772       85.85307416           NaN
773       57.71045218           NaN
```

```
[774 rows x 60 columns]
```

[4]:
```
VDem = pd.read_csv(VDem_url)
VDem = VDem[VDem.year != 2015]
```

1.3.2 Part b

The UN data contain certain rows that refer to groups of countries instead of to individual countries. Here's a list of these non-countries:

[5]:
```
noncountries = ['Arab World',  'Caribbean small states',  'Central Europe and␣
↪the Baltics',
   'Early-demographic dividend',  'East Asia & Pacific', 'East Asia & Pacific␣
↪(excluding high income)',
   'East Asia & Pacific (IDA & IBRD countries)', 'Euro area', 'Europe &␣
↪Central Asia',
   'Europe & Central Asia (excluding high income)', 'Europe & Central Asia␣
↪(IDA & IBRD countries)', 'European Union',
   'Fragile and conflict affected situations', 'Heavily indebted poor␣
↪countries (HIPC)',
   'High income', 'Late-demographic dividend', 'Latin America & Caribbean',
   'Latin America & Caribbean (excluding high income)',
   'Latin America & the Caribbean (IDA & IBRD countries)', 'Least developed␣
↪countries: UN classification',
   'Low & middle income', 'Low income', 'Lower middle income',
   'Middle East & North Africa', 'Middle East & North Africa (excluding high␣
↪income)',
   'Middle East & North Africa (IDA & IBRD countries)',
   'Middle income', 'North America', 'OECD members',
   'Other small states', 'Pacific island small states', 'Post-demographic␣
↪dividend',
   'Pre-demographic dividend', 'Small states', 'South Asia',
   'South Asia (IDA & IBRD)', 'Sub-Saharan Africa', 'Sub-Saharan Africa␣
↪(excluding high income)',
   'Sub-Saharan Africa (IDA & IBRD countries)', 'Upper middle income', 'World']
```

We can use `.query()` to remove the non-countries from the data, but in this case there are complications due to the space in the name of the column `Country Name` and the use of an external list. So here let's use an alternative method:

First, apply the `.isin(noncountries)` method to the `Country Name` column of the UN data to create a series of values that are `True` if the `Country Name` on a row is one of the non-countries, and `False` otherwise. Second, use the `~` operator to negate the logical values: turn `True` to `False` and vice versa. Finally, pass this logical series to the `.loc[]` attribute of the dataframe to drop the rows that refer to these noncountries from the UN data. (1 point)

(If you wanted to use `.query()`, you would first need to rename `Country Name` to remove the space, then you can use an `@` in front of `noncountries` to refer to the external list. But for this problem

follow the instructions listed above.)

[6]:
```
UN = UN.loc[~UN["Country Name"].isin(noncountries)]
UN
```

[6]:
```
                                  Series Name     Series Code  \
0                  Physicians (per 1,000 people)  SH.MED.PHYS.ZS
1                  Physicians (per 1,000 people)  SH.MED.PHYS.ZS
2                  Physicians (per 1,000 people)  SH.MED.PHYS.ZS
3                  Physicians (per 1,000 people)  SH.MED.PHYS.ZS
4                  Physicians (per 1,000 people)  SH.MED.PHYS.ZS
..                                           …              …
768  Health expenditure per capita (current US$)     SH.XPD.PCAP
769  Health expenditure per capita (current US$)     SH.XPD.PCAP
771  Health expenditure per capita (current US$)     SH.XPD.PCAP
772  Health expenditure per capita (current US$)     SH.XPD.PCAP
773  Health expenditure per capita (current US$)     SH.XPD.PCAP

              Country Name Country Code       1960 [YR1960] 1961 [YR1961]  \
0              Afghanistan          AFG  0.0348442494869232           NaN
1                  Albania          ALB   0.276291221380234           NaN
2                  Algeria          DZA   0.173148155212402           NaN
3           American Samoa          ASM                 NaN           NaN
4                  Andorra          ADO                 NaN           NaN
..                     …          …                   …             …
768  Virgin Islands (U.S.)          VIR                 NaN           NaN
769     West Bank and Gaza          WBG                 NaN           NaN
771            Yemen, Rep.          YEM                 NaN           NaN
772                 Zambia          ZMB                 NaN           NaN
773               Zimbabwe          ZWE                 NaN           NaN

    1962 [YR1962] 1963 [YR1963] 1964 [YR1964]       1965 [YR1965]  …  \
0             NaN           NaN           NaN  0.0634277984499931  …
1             NaN           NaN           NaN    0.48128342628479  …
2             NaN           NaN           NaN   0.116413652896881  …
3             NaN           NaN           NaN                 NaN  …
4             NaN           NaN           NaN                 NaN  …
..            …             …             …                   …  …
768           NaN           NaN           NaN                 NaN  …
769           NaN           NaN           NaN                 NaN  …
771           NaN           NaN           NaN                 NaN  …
772           NaN           NaN           NaN                 NaN  …
773           NaN           NaN           NaN                 NaN  …

    2006 [YR2006] 2007 [YR2007] 2008 [YR2008] 2009 [YR2009] 2010 [YR2010]  \
0           0.136         0.146         0.145         0.175         0.194
1            1.15         1.146           NaN         1.144         1.132
```

```
2              NaN          1.207            NaN            NaN          1.207
3              NaN            NaN            NaN            NaN            NaN
4             3.64          3.716            NaN          3.912              4
..             …              …              …              …              …
768            NaN            NaN            NaN            NaN            NaN
769            NaN            NaN            NaN            NaN            NaN
771     52.0957528    58.08815663     69.6851483    65.91392692    67.74874258
772    62.92278027     48.1756618    66.53854373    53.65967398    64.17510388
773    21.24877429    17.80401666    16.21371054    37.20702422    36.36279411

     2011 [YR2011] 2012 [YR2012] 2013 [YR2013] 2014 [YR2014] 2015 [YR2015]
0            0.234          0.225          0.266            NaN            NaN
1            1.113          1.145          1.145            NaN            NaN
2              NaN            NaN            NaN            NaN            NaN
3              NaN            NaN            NaN            NaN            NaN
4              NaN            NaN            NaN            NaN            NaN
..             …              …              …              …              …
768            NaN            NaN            NaN            NaN            NaN
769            NaN            NaN            NaN            NaN            NaN
771    64.65160379    73.90669273    78.52249036    79.93696624            NaN
772    70.52581827    82.86919808    87.83302346    85.85307416            NaN
773    48.46958014    57.25376348    62.30922835    57.71045218            NaN

[651 rows x 60 columns]
```

1.3.3 Part c

Reshape the UN data to move the years from the columns to the rows. (Once the years are in the rows, they will have values such as "1960 [YR1960]".) (2 points)

[7]:
```
UN_clean = pd.melt(UN, id_vars = ['Series Name', 'Series Code', 'Country Name',␣
↪'Country Code'])
UN_clean
```

[7]:
```
                                        Series Name     Series Code  \
0                       Physicians (per 1,000 people)  SH.MED.PHYS.ZS
1                       Physicians (per 1,000 people)  SH.MED.PHYS.ZS
2                       Physicians (per 1,000 people)  SH.MED.PHYS.ZS
3                       Physicians (per 1,000 people)  SH.MED.PHYS.ZS
4                       Physicians (per 1,000 people)  SH.MED.PHYS.ZS
…                                                 …              …
36451  Health expenditure per capita (current US$)     SH.XPD.PCAP
36452  Health expenditure per capita (current US$)     SH.XPD.PCAP
36453  Health expenditure per capita (current US$)     SH.XPD.PCAP
36454  Health expenditure per capita (current US$)     SH.XPD.PCAP
36455  Health expenditure per capita (current US$)     SH.XPD.PCAP
```

```
                Country Name Country Code       variable                value
0                Afghanistan          AFG  1960 [YR1960]  0.0348442494869232
1                    Albania          ALB  1960 [YR1960]   0.276291221380234
2                    Algeria          DZA  1960 [YR1960]   0.173148155212402
3             American Samoa          ASM  1960 [YR1960]                 NaN
4                    Andorra          ADO  1960 [YR1960]                 NaN
…                        …          …            …                    …
36451  Virgin Islands (U.S.)          VIR  2015 [YR2015]                 NaN
36452     West Bank and Gaza          WBG  2015 [YR2015]                 NaN
36453            Yemen, Rep.          YEM  2015 [YR2015]                 NaN
36454                 Zambia          ZMB  2015 [YR2015]                 NaN
36455               Zimbabwe          ZWE  2015 [YR2015]                 NaN

[36456 rows x 6 columns]
```

1.3.4 Part d

Rename the `variable` column to `year`. Then use string methods to remove the ends such as "[YR1960]" from the values of the new `year` column and convert the column to an integer data type.

Also, for whatever reason, real world data often contains multiple variables that are just different representations of the same information. In this case, the `Series Name` and `Series Code` variables tell us exactly the same thing, and the `Country Name` and `Country Code` variables tell us exactly the same thing. Unless I have a very good reason to keep both, I generally prefer to drop variables that are redundant and coded in a less helpful way. So drop `Series Code` and `Country Code`. (2 points)

[8]:
```
UN_clean = UN_clean.rename(columns={'variable':'year'})
UN_clean.year = UN_clean.year.str.split(' ').str[0].astype(int)
UN_clean = UN_clean.drop(['Series Code','Country Code'], axis=1)
UN_clean
```

[8]:
```
                                      Series Name           Country Name  \
0                 Physicians (per 1,000 people)            Afghanistan
1                 Physicians (per 1,000 people)                Albania
2                 Physicians (per 1,000 people)                Algeria
3                 Physicians (per 1,000 people)         American Samoa
4                 Physicians (per 1,000 people)                Andorra
…                                           …                      …
36451  Health expenditure per capita (current US$)  Virgin Islands (U.S.)
36452  Health expenditure per capita (current US$)     West Bank and Gaza
36453  Health expenditure per capita (current US$)             Yemen, Rep.
36454  Health expenditure per capita (current US$)                  Zambia
36455  Health expenditure per capita (current US$)                Zimbabwe

       year               value
0      1960  0.0348442494869232
```

```
1       1960   0.276291221380234
2       1960   0.173148155212402
3       1960                 NaN
4       1960                 NaN
…       …                    …
36451   2015                 NaN
36452   2015                 NaN
36453   2015                 NaN
36454   2015                 NaN
36455   2015                 NaN

[36456 rows x 4 columns]
```

1.3.5 Part e

Reshape the data to move the values of `Series Name` to separate columns. Make sure all of the columns exist in the dataframe after reshaping and are not stored in a row index or multi-index. Then rename the columns so that all of the columns have concise and descriptive names. (2 points)

[9]:
```
UN_clean = UN_clean.pivot_table(index=['Country Name', 'year'],columns='Series␣
 ↪Name',values='value',aggfunc='first')
UN_clean = pd.DataFrame(UN_clean.to_records())
UN_clean = UN_clean.rename(columns={'Country Name':'country_name',
                           'year':'year',
                           'Health expenditure per capita (current US$)':
 ↪'HealthCost',
                           'Physicians (per 1,000 people)':'DocPer1000',
                           'Prevalence of undernourishment (% of population)':
 ↪'UnderNourish%'})
UN_clean
```

[9]:
```
     country_name  year   HealthCost          DocPer1000 UnderNourish%
0     Afghanistan  1960          NaN  0.0348442494869232           NaN
1     Afghanistan  1965          NaN  0.0634277984499931           NaN
2     Afghanistan  1970          NaN  0.0649000033736229           NaN
3     Afghanistan  1981          NaN  0.0769999995827675           NaN
4     Afghanistan  1986          NaN   0.183100000023842           NaN
…             …    …           …                   …            …
6396     Zimbabwe  2011  48.46958014               0.083         33.5
6397     Zimbabwe  2012  57.25376348                 NaN         33.2
6398     Zimbabwe  2013  62.30922835                 NaN         33.5
6399     Zimbabwe  2014  57.71045218                 NaN           34
6400     Zimbabwe  2015          NaN                 NaN         33.4

[6401 rows x 5 columns]
```

1.3.6 Part f

Next we are going to join the cleaned UN data with the VDem data. In a perfect world, both datasets would include a shared numeric country ID field that we can use to match countries in one dataset to countries in the other. Unfortunately the UN data identifies the countries only by name. Worse still, while there is a big overlap the two datasets cover different sets of countries.

First decide whether this merge is a one-to-one, one-to-many, many-to-one, or many-to-many merge and describe your rationale in words.

Then perform a test merge that checks whether your expectation that the merge is one-to-one, one-to-many, many-to-one, or many-to-many is confirmed, and reports whether each row is matched, appears only in the UN data, or appears only in the VDem data. Use the `.unique()` or `.value_counts()` method to display the names of the countries that are not matched. (2 points)

One to one becuase there is one entry for each country per year in the UN datafarme and one entry for each country per year in the VDem dataframe

[10]:
```
merged = pd.merge(UN_clean,VDem, on=['country_name','year'],
                      how='outer',
                      validate='one_to_one',
                      indicator='matched')
merged['matched'].value_counts()
```

[10]:
```
both          4606
right_only    3852
left_only     1795
Name: matched, dtype: int64
```

1.3.7 Part g

There are many unmatched rows in this merge. There are three reasons why rows failed to match: * Differences in geographical coverage: for example, the VDem data includes Taiwan, but the UN data does not * Differences in time coverage: for example, the UN data includes records for France every year from 1970 through 2014, and VDem includes rows for France from 1960 to 2012, leaving 12 rows for France without matching years * Differences in spelling: for example, South Korea is called “Korea, Rep.” in the UN data and “Korea_South” in the VDem data.

We can’t do anything about differences in geographic or temporal coverage. But we can recode some country names to account for differences in spelling and to match more rows that should match. Here is a list of differently spelled countries:

- “Burma_Myanmar” in VDem is “Myanmar” in the UN data
- “Cape Verde” in VDem is “Cabo Verde” in the UN data
- “Congo_Democratic Republic of” in VDem is “Congo, Dem. Rep.” in the UN data
- “Congo_Republic of the” in VDem is “Congo, Rep.” in the UN data
- “East Timor” in VDem is “Timor-Leste” in the UN data
- “Egypt” in VDem is “Egypt, Arab Rep.” in the UN data
- “Gambia” in VDem is “Gambia, The” in the UN data
- “Iran” in VDem is “Iran, Islamic Rep.” in the UN data
- “Ivory Coast” in VDem is “Cote d’Ivoire” in the UN data

- "Korea_North" in VDem is "Korea, Dem. People's Rep." in the UN data
- "Korea_South" in VDem is "Korea, Rep." in the UN data
- "Kyrgyzstan" in VDem is "Kyrgyz Republic" in the UN data
- "Laos" in VDem is "Lao PDR" in the UN data
- "Macedonia" in VDem is "Macedonia, FYR" in the UN data
- "Palestine_West_Bank" in VDem is "West Bank and Gaza" in the UN Data (there is also "Palestine_Gaza" in VDem, but since the UN combines data for the West Bank and Gaza, let's just use "Palestine_West_Bank" for this assignment)
- "Russia" in VDem is "Russian Federation" in the UN data
- "Slovakia" in VDem is "Slovak Republic" in the UN data
- "Syria" in VDem is "Syrian Arab Republic" in the UN data
- "Venezuela" in VDem is "Venezuela, RB" in the UN data
- "Vietnam_Democratic Republic of" in VDem is "Vietnam" in the UN data
- "Yemen" in VDem is "Yemen, Rep." in the UN data

Recode the country names listed above in one of the two dataframes to match the names in the other dataframe. Then perform an inner join of the two dataframes. Some rows will be dropped because of differences in coverage, but no rows will be dropped because of differences in spelling. (2 points)

[11]:
```
UN_clean.loc[(UN_clean.country_name =="Myanmar"),'country_name']="Burma_Myanmar"
UN_clean.loc[(UN_clean.country_name =="Cabo Verde"),'country_name']="Cape Verde"
UN_clean.loc[(UN_clean.country_name =="Congo, Dem. Rep."),'country_name']="Cape␣
 ↪Verde"
UN_clean.loc[(UN_clean.country_name =="Congo, Rep.
 ↪"),'country_name']="Congo_Democratic Republic of"
UN_clean.loc[(UN_clean.country_name =="Congo, Dem. Rep.
 ↪"),'country_name']="Congo_Republic of the"
UN_clean.loc[(UN_clean.country_name =="Timor-Leste"),'country_name']="East␣
 ↪Timor"
UN_clean.loc[(UN_clean.country_name =="Egypt, Arab Rep.
 ↪"),'country_name']="Egypt"
UN_clean.loc[(UN_clean.country_name =="Gambia, The"),'country_name']="Gambia"
UN_clean.loc[(UN_clean.country_name =="Iran, Islamic Rep.
 ↪"),'country_name']="Iran"
UN_clean.loc[(UN_clean.country_name =="Cote d'Ivoire"),'country_name']="Ivory␣
 ↪Coast"
UN_clean.loc[(UN_clean.country_name =="Korea, Dem. People's Rep.
 ↪"),'country_name']="Korea_North"
UN_clean.loc[(UN_clean.country_name =="Korea, Rep.
 ↪"),'country_name']="Korea_South"
UN_clean.loc[(UN_clean.country_name =="Kyrgyz␣
 ↪Republic"),'country_name']="Kyrgyzstan"
UN_clean.loc[(UN_clean.country_name =="Lao PDR"),'country_name']="Laos"
UN_clean.loc[(UN_clean.country_name =="Macedonia,␣
 ↪FYR"),'country_name']="Macedonia"
```

```
UN_clean.loc[(UN_clean.country_name =="West Bank and␣
 ↪Gaza"),'country_name']="Palestine_West_Bank"
UN_clean.loc[(UN_clean.country_name =="Russian␣
 ↪Federation"),'country_name']="Russia"
UN_clean.loc[(UN_clean.country_name =="Slovak␣
 ↪Republic"),'country_name']="Slovakia"
UN_clean.loc[(UN_clean.country_name =="Syrian Arab␣
 ↪Republic"),'country_name']="Syria"
UN_clean.loc[(UN_clean.country_name =="Venezuela,␣
 ↪RB"),'country_name']="Venezuela"
UN_clean.loc[(UN_clean.country_name␣
 ↪=="Vietnam"),'country_name']="Vietnam_Democratic Republic of"
UN_clean.loc[(UN_clean.country_name =="Yemen, Rep."),'country_name']="Yemen"
```

[12]:
```
merged = pd.merge(UN_clean,VDem, on=['country_name','year'],
                  how='inner')
merged
```

[12]:

	country_name	year	HealthCost	DocPer1000	UnderNourish%	X1	\
0	Afghanistan	1960	NaN	0.0348442494869232	NaN	1583	
1	Afghanistan	1965	NaN	0.0634277984499931	NaN	1588	
2	Afghanistan	1970	NaN	0.0649000033736229	NaN	1593	
3	Afghanistan	1981	NaN	0.0769999995827675	NaN	1604	
4	Afghanistan	1986	NaN	0.183100000023842	NaN	1609	
…	…	…	…	…	…	…	
5143	Zimbabwe	2010	36.36279411	0.068	34.7	3035	
5144	Zimbabwe	2011	48.46958014	0.083	33.5	3036	
5145	Zimbabwe	2012	57.25376348	NaN	33.2	3037	
5146	Zimbabwe	2013	62.30922835	NaN	33.5	3038	
5147	Zimbabwe	2014	57.71045218	NaN	34	3039	

	country_id	country_text_id	historical_date	codingstart	…	\
0	36	AFG	1960-01-01	1900	…	
1	36	AFG	1965-01-01	1900	…	
2	36	AFG	1970-01-01	1900	…	
3	36	AFG	1981-01-01	1900	…	
4	36	AFG	1986-01-01	1900	…	
…	…	…	…	…	…	
5143	62	ZWE	2010-01-01	1900	…	
5144	62	ZWE	2011-01-01	1900	…	
5145	62	ZWE	2012-01-01	1900	…	
5146	62	ZWE	2013-01-01	1900	…	
5147	62	ZWE	2014-01-01	1900	…	

	v2xcs_ccsi_codehigh	v2xcs_ccsi_codelow	v2xps_party	\
0	0.415384	0.143550	0.074516	
1	0.615534	0.288714	0.177830	

```
2                0.600535            0.271911     0.177830
3                0.147095            0.023951     0.143448
4                0.147095            0.023951     0.143448
…                   …                   …          …
5143             0.722976            0.432416     0.438434
5144             0.722976            0.432416     0.438434
5145             0.722976            0.432416     0.438434
5146             0.631608            0.304144     0.443507
5147             0.624166            0.292913     0.443507

      v2xps_party_codehigh  v2xps_party_codelow  v2x_gender  \
0                 0.162687             0.028557    0.181335
1                 0.304231             0.090927    0.215910
2                 0.304231             0.090927    0.211898
3                 0.254819             0.070712    0.209011
4                 0.254819             0.070712    0.209011
…                    …                    …          …
5143              0.582274             0.302371    0.559720
5144              0.582274             0.302371    0.559720
5145              0.582274             0.302371    0.559720
5146              0.601263             0.294328         NaN
5147              0.601263             0.294328         NaN

      v2x_gender_codehigh  v2x_gender_codelow  v2x_gencl  v2x_gencl_codehigh
0                0.232855            0.129815   0.172381            0.301402
1                0.277255            0.154566   0.201414            0.350518
2                0.268672            0.155124   0.201414            0.350518
3                0.273436            0.144586   0.222300            0.369985
4                0.273436            0.144586   0.222300            0.369985
…                   …                   …          …                   …
5143             0.641812            0.477629   0.459267            0.623338
5144             0.641812            0.477629   0.459267            0.623338
5145             0.641812            0.477629   0.459267            0.623338
5146                  NaN                 NaN   0.508582            0.665112
5147                  NaN                 NaN   0.508582            0.665112

[5148 rows x 103 columns]
```

1.4 Problem 2

Kickstarter is a website in which people can pledge financial support for creative projects. Patrons are only charged if a project raises enough money to meet a pre-specified goal, and projects can offer items as "rewards" for patrons who contribute at particular levels. One interesting aspect of Kickstarter is the ability to search projects by "ending soon". If you have a few dollars to spare and want to feel like a hero, you can swoop in at the last minute to contribute enough for a project to meet its goal.

Cathie So created a project on Kaggle in which she scraped Kickstarter and collected data on 4000

live projects (projects that were currently collecting pledges from patrons) as of October 10, 2016, at 5pm Pacific time. The data are here:

[13]:
```
kickstarter = pd.read_csv("https://github.com/jkropko/DS-6001/raw/master/
↪localdata/live.csv")
kickstarter
```

[13]:
```
      Unnamed: 0  amt.pledged  \
0              0      15823.0
1              1       6859.0
2              2      17906.0
3              3      67081.0
4              4      32772.0
…            …            …
3995        3995       4403.0
3996        3996       1304.0
3997        3997          1.0
3998        3998         10.0
3999        3999         35.0

                                                  blurb  \
0     \n'Catalysts, Explorers & Secret Keepers: Wome…
1     \nA unique handmade picture book for kids & ar…
2     \nA horror comedy about a repairman who was in…
3     \nThe Johnny Wander autobio omnibus you've all…
4     \nThe vision for this project is the establish…
…                                                   …
3995  \nEARTH IS BUT ONE FRUIT ON THE TREE OF LIFE. …
3996  \nImagine designing an item with an easy-to-us…
3997  \nUnique themed London venue and hostel for 9g…
3998                          \nAll in One Phone Case\n
3999  \nLuxury Sunglasses built with Titanium, Carbo…

                                    by country currency  \
0              Museum of Science Fiction      US      usd
1       Tyrone Wells & Broken Eagle, LLC      US      usd
2                          Tessa Stone       US      usd
3                        Johnny Wander       US      usd
4     Beau's All Natural Brewing Company      RW      cad
…                                     …     …       …
3995                       Lewis Brown       US      usd
3996                  Your Expressions       US      usd
3997                    Martin Wojtala       GB      gbp
3998              All in One Phone Case      US      usd
3999                       Carlos Araujo      US      usd

                     end.time           location  percentage.funded  \
```

```
0     2016-11-01T23:59:00-04:00     Washington, DC                186
1     2016-11-25T01:13:33-05:00       Portland, OR                  8
2     2016-11-23T23:00:00-05:00    Los Angeles, CA                102
3     2016-11-01T23:50:00-04:00       Brooklyn, NY                191
4     2016-11-18T23:05:48-05:00     Kigali, Rwanda                 34
…                           …               …                …
3995  2016-11-20T01:10:00-05:00         Denver, CO                 88
3996  2016-11-15T16:00:00-05:00  San Francisco, CA                  5
3997  2016-10-30T09:36:06-04:00         London, UK                  0
3998  2016-11-17T12:11:26-05:00    Tallahassee, FL                  0
3999  2016-12-11T00:11:01-05:00       New York, NY                  0

                state                                              title  \
0                  DC  Catalysts, Explorers & Secret Keepers: Women o…
1                  OR  The Whatamagump (a hand-crafted story picture …
2                  CA                          Not Drunk Enough Volume 1!
3                  NY  Our Cats Are More Famous Than Us: A Johnny Wan…
4     Kigali Province                   The Rwanda Craft Brewery Project
…               …                                                   …
3995               CO                                BROWN HORNET OMNIVERSE
3996               CA      3D Pixie - App to Design Personalized Jewelry
3997          England                                        9HUB - London
3998               FL                                 All in One Phone Case
3999               NY  Edward & Lux- Classic & Timeless Aviator Sungl…

         type                                                  url
0       Town  /projects/1608905146/catalysts-explorers-and-s…
1       Town  /projects/thewhatamagump/the-whatamagump-a-han…
2       Town  /projects/1890925998/not-drunk-enough-volume-1…
3     County  /projects/746734715/our-cats-are-more-famous-t…
4       Town  /projects/beaus/the-rwanda-craft-brewery-proje…
…       …                                                 …
3995    Town  /projects/brownhornetomni/brown-hornet-omniver…
3996    Town  /projects/yourexpressions/3d-pixie-app-to-crea…
3997    Town     /projects/1132099243/9hub-london?ref=discovery
3998    Town  /projects/203104559/all-in-one-phone-case?ref=…
3999    Town  /projects/1833705733/edward-and-lux-classic-an…

[4000 rows x 13 columns]
```

1.4.1 Part a

Notice that the `end.time` column, the date and time at which the project stops accepting pledges, is formatted as follows:

`2016-11-01T23:59:00-04:00`

This formatting is "YYYY-MM-DDThh:mm:ss-TZD": four digits for the year, a dash, two digits for the month, another dash, and two digits for the day; the "T" separates the dates from the time;

two digits for the hour, minute and second, separated by colons; and the time zone expressed as hours difference from Greenwich mean time (also called UTC), and -04:00 is four hours earlier than UTC, for example.

But `end.time` is also currently read as a string, with `object` data type:

[14]:
```
kickstarter.dtypes
```

[14]:
```
Unnamed: 0             int64
amt.pledged          float64
blurb                 object
by                    object
country               object
currency              object
end.time              object
location              object
percentage.funded      int64
state                 object
title                 object
type                  object
url                   object
dtype: object
```

Convert `end.time` to a timestamp, and extract the month, day, year, hour, minute, and second of the end time. To allow the `pd.to_datetime()` function to read timezones, use the `utc=True` argument. (2 points)

[15]:
```
kickstarter['end.time'] = pd.to_datetime(kickstarter['end.time'],utc=True)
endtimes = kickstarter['end.time']
kickstarter['month'] = [x.month for x in endtimes]
kickstarter['day'] = [x.day for x in endtimes]
kickstarter['year'] = [x.year for x in endtimes]
kickstarter['hour'] = [x.hour for x in endtimes]
kickstarter['minute'] = [x.minute for x in endtimes]
kickstarter['second'] = [x.second for x in endtimes]
kickstarter
```

[15]:
```
      Unnamed: 0  amt.pledged  \
0              0      15823.0
1              1       6859.0
2              2      17906.0
3              3      67081.0
4              4      32772.0
…            …           …
3995        3995       4403.0
3996        3996       1304.0
3997        3997          1.0
3998        3998         10.0
3999        3999         35.0
```

```
                                                  blurb  \
0     \n'Catalysts, Explorers & Secret Keepers: Wome…
1     \nA unique handmade picture book for kids & ar…
2     \nA horror comedy about a repairman who was in…
3     \nThe Johnny Wander autobio omnibus you've all…
4     \nThe vision for this project is the establish…
…                                                   …
3995  \nEARTH IS BUT ONE FRUIT ON THE TREE OF LIFE. …
3996  \nImagine designing an item with an easy-to-us…
3997  \nUnique themed London venue and hostel for 9g…
3998                          \nAll in One Phone Case\n
3999  \nLuxury Sunglasses built with Titanium, Carbo…

                                    by country currency  \
0              Museum of Science Fiction      US      usd
1        Tyrone Wells & Broken Eagle, LLC      US      usd
2                           Tessa Stone      US      usd
3                         Johnny Wander      US      usd
4     Beau's All Natural Brewing Company      RW      cad
…                                     …     …       …
3995                        Lewis Brown      US      usd
3996                  Your Expressions       US      usd
3997                    Martin Wojtala       GB      gbp
3998              All in One Phone Case      US      usd
3999                       Carlos Araujo      US      usd

                      end.time           location  percentage.funded  \
0    2016-11-02 03:59:00+00:00     Washington, DC                186
1    2016-11-25 06:13:33+00:00       Portland, OR                  8
2    2016-11-24 04:00:00+00:00    Los Angeles, CA                102
3    2016-11-02 03:50:00+00:00       Brooklyn, NY                191
4    2016-11-19 04:05:48+00:00     Kigali, Rwanda                 34
…                          …                  …                 …
3995 2016-11-20 06:10:00+00:00         Denver, CO                 88
3996 2016-11-15 21:00:00+00:00  San Francisco, CA                  5
3997 2016-10-30 13:36:06+00:00         London, UK                  0
3998 2016-11-17 17:11:26+00:00    Tallahassee, FL                  0
3999 2016-12-11 05:11:01+00:00       New York, NY                  0

                state                                              title  \
0                  DC  Catalysts, Explorers & Secret Keepers: Women o…
1                  OR  The Whatamagump (a hand-crafted story picture …
2                  CA                          Not Drunk Enough Volume 1!
3                  NY  Our Cats Are More Famous Than Us: A Johnny Wan…
4     Kigali Province                    The Rwanda Craft Brewery Project
…                 …                                                 …
```

```
3995              CO                                BROWN HORNET OMNIVERSE
3996              CA      3D Pixie - App to Design Personalized Jewelry
3997         England                                           9HUB - London
3998              FL                                 All in One Phone Case
3999              NY  Edward & Lux- Classic & Timeless Aviator Sungl…

        type                                                  url  month  day  \
0       Town  /projects/1608905146/catalysts-explorers-and-s…     11    2
1       Town  /projects/thewhatamagump/the-whatamagump-a-han…     11   25
2       Town  /projects/1890925998/not-drunk-enough-volume-1…     11   24
3     County  /projects/746734715/our-cats-are-more-famous-t…     11    2
4       Town  /projects/beaus/the-rwanda-craft-brewery-proje…     11   19
…        …                                                …    …   …
3995    Town  /projects/brownhornetomni/brown-hornet-omniver…     11   20
3996    Town  /projects/yourexpressions/3d-pixie-app-to-crea…     11   15
3997    Town     /projects/1132099243/9hub-london?ref=discovery     10   30
3998    Town  /projects/203104559/all-in-one-phone-case?ref=…     11   17
3999    Town  /projects/1833705733/edward-and-lux-classic-an…     12   11

      year  hour  minute  second
0     2016     3      59       0
1     2016     6      13      33
2     2016     4       0       0
3     2016     3      50       0
4     2016     4       5      48
…    …   …      …      …
3995  2016     6      10       0
3996  2016    21       0       0
3997  2016    13      36       6
3998  2016    17      11      26
3999  2016     5      11       1

[4000 rows x 19 columns]
```

1.4.2 Part b

Create a dataframe with one row for every ending day in the `kickstarter` data that reports the average amount pledged (`amt.pledged`) on each day. Sort the rows in descending order by average amount pledged, and display the five days with the highest averages. (2 points)

[16]:
```
df = kickstarter.groupby(['month','day']).agg({'amt.pledged':['mean']}).
↪reset_index()
df.columns = ['month','day','pledgedmean']
df.sort_values('pledgedmean',ascending=False)[0:5]
```

[16]:
```
    month  day   pledgedmean
46     12   14  47938.375000
```

```
6      11    4  26975.388889
13     11   11  24990.669065
49     12   17  22160.230769
20     11   18  21016.234043
```

1.4.3 Part c

Display the text of the longest `blurb` in the data. (2 points)

[17]:
```
kickstarter.blurb[kickstarter.blurb.str.len().sort_values(ascending=False).
↪index[0]]
```

[17]:
```
'\nWe are charismatic anti-rock band hailing from Winnipeg, Manitoba and we are
determined to release a debut album by the summer of 2017!\n'
```

1.4.4 Part d

How many blurbs for projects with end dates between November 15, 2016 and December 7, 2016 contain the phrase "science fiction"? [Hint: Don't forget to make this search case-insensitive and to sort the `kickstarter` dataframe by `end.time` before setting `end.time` as the index.] (2 points)

[26]:
```
df = kickstarter.sort_values('end.time',ascending=False)
df.index = df['end.time']
df = df['11/15/2016':'12/7/2016']
df = df[df.blurb.str.lower().str.contains('science fiction')]
len(df)
```

[26]:
```
6
```

[ ]:
```

```