Assignment 1

September 30, 2020

---

*You are currently looking at **version 1.1** of this notebook. To download notebooks and datafiles, as well as get help on Jupyter notebooks in the Coursera platform, visit the Jupyter Notebook FAQ course resource.*

---

1 Assignment 1

In this assignment, you'll be working with messy medical data and using regex to extract relevant infromation from the data.

Each line of the `dates.txt` file corresponds to a medical note. Each note has a date that needs to be extracted, but each date is encoded in one of many formats.

The goal of this assignment is to correctly identify all of the different date variants encoded in this dataset and to properly normalize and sort the dates.

Here is a list of some of the variants you might encounter in this dataset: * 04/20/2009; 04/20/09; 4/20/09; 4/3/09 * Mar-20-2009; Mar 20, 2009; March 20, 2009; Mar. 20, 2009; Mar 20 2009; * 20 Mar 2009; 20 March 2009; 20 Mar. 2009; 20 March, 2009 * Mar 20th, 2009; Mar 21st, 2009; Mar 22nd, 2009 * Feb 2009; Sep 2009; Oct 2010 * 6/2008; 12/2009 * 2009; 2010

Once you have extracted these date patterns from the text, the next step is to sort them in ascending chronological order accoring to the following rules: * Assume all dates in xx/xx/xx format are mm/dd/yy * Assume all dates where year is encoded in only two digits are years from the 1900's (e.g. 1/5/89 is January 5th, 1989) * If the day is missing (e.g. 9/2009), assume it is the first day of the month (e.g. September 1, 2009). * If the month is missing (e.g. 2010), assume it is the first of January of that year (e.g. January 1, 2010). * Watch out for potential typos as this is a raw, real-life derived dataset.

With these rules in mind, find the correct date in each note and return a pandas Series in chronological order of the original Series' indices.

For example if the original series was this:

```
0    1999
1    2010
2    1978
3    2015
4    1985
```

Your function should return this:

```
0    2
1    4
2    0
3    1
4    3
```

Your score will be calculated using Kendall's tau, a correlation measure for ordinal data.
This function should return a Series of length 500 and dtype int.

```
In [93]: import pandas as pd
         import numpy as np
         import re

         doc = []
         with open('dates.txt') as file:
             for line in file:
                 doc.append(line)

         df = pd.DataFrame(doc, columns = ['text'])
```

```
In [144]: def date_sorter():

              f1 = df['text'].str.extract(r'(?P<date>\d{1,2}[/|-]\d{1,2}[/|-][1,2]?\d?\d{2})')

              f2 = df['text'].str.extract(r'(?P<date>(?:\d{,2}\s)?(?:Jan|Feb|Mar|Apr|May|Jun|Jul

              f3 = df['text'].str.extract(r'(?P<date>\d{0,2}/?[1,2]\d{3})')

              data = pd.to_datetime(f1.fillna(f2).fillna(f3).str.replace('Decemeber','December')

              data = data.sort_values(ascending = True)

              return pd.Series(data.index)

          date_sorter()
```

```
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:1: FutureWarning: currently extract
  """Entry point for launching an IPython kernel.
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:3: FutureWarning: currently extract
  This is separate from the ipykernel package so we can avoid doing imports until
/opt/conda/lib/python3.6/site-packages/ipykernel_launcher.py:5: FutureWarning: currently extract
  """
```

```
Out[144]: 0        9
          1       84
          2        2
          3       53
          4       28
```

```
5       474
6       153
7        13
8       129
9        98
10      111
11      225
12       31
13      171
14      191
15      486
16      335
17      415
18       36
19      405
20      323
21      422
22      375
23      380
24      345
25       57
26      481
27      436
28      104
29      299
       ...
470     220
471     208
472     243
473     139
474     320
475     383
476     244
477     286
478     480
479     431
480     279
481     198
482     381
483     463
484     366
485     439
486     255
487     401
488     475
489     257
490     152
491     235
```

```
492    464
493    253
494    427
495    231
496    141
497    186
498    161
499    413
Length: 500, dtype: int64
```

def date_sorter():

```
month1 = {'Jan':'01','Feb':'02','Mar':'03','Apr':'04','May':'05','Jun':'06',
             'Jul':'07','Aug':'08','Sep':'09','Oct':'10','Nov':'11','Dec':'12'}

month2 = {'January':'01','February':'02','March':'03','April':'04','May':'05','June':'06',
             'July':'07','August':'08','September':'09','October':'10','November':'11','December

f1 = df['text'].str.extractall(r'(?P<year>\W[^/][12]\d{3}\W[^/])')
f1['year']= f1['year'].str.replace('[^0-9]','')
f1['year'] = f1['year'].apply(lambda x: x.strip())
f1['month'] = '01'
f1['day'] = '01'
f1['data1'] = f1['year']+'-'+f1['month']+'-'+f1['day']
f1 = f1.reset_index(level=[0,1]).drop(['match'],axis=1).rename(columns = {'level_0':'id'})

f2 = df['text'].str.extractall(r'(?P<date>(?P<day>\W?\d\d\W?)\W(?P<month>\W?Jan\W?|\W?Feb\W?|\W?
f2['day'] = f2['day'].str.replace('[\W]','')
f2['month'] = f2['month'].str.replace('[\W]','')
f2['year'] = f2['year'].str.replace('[\W]','')
f2['day'] = f2['day'].apply(lambda x: x.strip())
f2['month'] = f2['month'].apply(lambda x: x.strip())
f2['year'] = f2['year'].apply(lambda x: x.strip())
f2['month'] = f2['month'].map(month1)
f2['data2'] = f2['year']+'-'+f2['month']+'-'+f2['day']
f2 = f2.reset_index(level=[0,1]).drop(['match','date'],axis=1).rename(columns = {'level_0':'id'}

f3 = df['text'].str.extractall(r'(?P<date>(?P<month>\W?Jan\W?|\W?Feb\W?|\W?Mar\W?|\W?Apr\W?|\W?M
f3['day'] = f3['day'].str.replace('[\W]','')
f3['month'] = f3['month'].str.replace('[\W]','')
f3['year'] = f3['year'].str.replace('[\W]','')
f3['day'] = f3['day'].apply(lambda x: x.strip())
f3['month'] = f3['month'].apply(lambda x: x.strip())
f3['year'] = f3['year'].apply(lambda x: x.strip())
f3['month'] = f3['month'].map(month1)
f3['data3'] = f3['year']+'-'+f3['month']+'-'+f3['day']
f3 = f3.reset_index(level=[0,1]).drop(['match','date'],axis=1).rename(columns = {'level_0':'id'}
```

```
f4 = df['text'].str.extractall(r'(?P<date>(?P<month>\W?Jan\W?|\W?Feb\W?|\W?Mar\W?|\W?Apr\W?|\W?M
f4['month'] = f4['month'].str.replace('[\W]','')
f4['year'] = f4['year'].str.replace('[\W]','')
f4['month'] = f4['month'].apply(lambda x: x.strip())
f4['year'] = f4['year'].apply(lambda x: x.strip())
f4['day'] = '01'
f4['month'] = f4['month'].map(month1)
f4['data4'] = f4['year']+'-'+f4['month']+'-'+f4['day']
f4 = f4.reset_index(level=[0,1]).drop(['match','date'],axis=1).rename(columns = {'level_0':'id'}

f5 = df['text'].str.extractall(r'(?P<date>(?P<month>\W?January\W?|\W?February\W?|\W?March\W?|\W?
f5['day'] = f5['day'].str.replace('[\W]','')
f5['month'] = f5['month'].str.replace('[\W]','')
f5['year'] = f5['year'].str.replace('[\W]','')
f5['day'] = f5['day'].apply(lambda x: x.strip())
f5['month'] = f5['month'].apply(lambda x: x.strip())
f5['year'] = f5['year'].apply(lambda x: x.strip())
f5['month'] = f5['month'].map(month2)
f5['data5'] = f5['year']+'-'+f5['month']+'-'+f5['day']
f5 = f5.reset_index(level=[0,1]).drop(['match','date'],axis=1).rename(columns = {'level_0':'id'}

f6 = df['text'].str.extractall(r'(?P<date>(?P<month>\W?January\W?|\W?February\W?|\W?March\W?|\W?
f6['month'] = f6['month'].str.replace('[\W]','')
f6['year'] = f6['year'].str.replace('[\W]','')
f6['month'] = f6['month'].apply(lambda x: x.strip())
f6['year'] = f6['year'].apply(lambda x: x.strip())
f6['month'] = f6['month'].map(month2)
f6['day'] = '01'
f6['data6'] = f6['year']+'-'+f6['month']+'-'+f6['day']
f6 = f6.reset_index(level=[0,1]).drop(['match','date'],axis=1).rename(columns = {'level_0':'id'}

f7 = df['text'].str.extractall(r'(?P<date>(?P<day>\W?\d\d\W?)\W(?P<month>\W?January\W?|\W?Februa
f7['day'] = f7['day'].str.replace('[\W]','')
f7['month'] = f7['month'].str.replace('[\W]','')
f7['year'] = f7['year'].str.replace('[\W]','')
f7['day'] = f7['day'].apply(lambda x: x.strip())
f7['month'] = f7['month'].apply(lambda x: x.strip())
f7['year'] = f7['year'].apply(lambda x: x.strip())
f7['month'] = f7['month'].map(month2)
f7['data7'] = f7['year']+'-'+f7['month']+'-'+f7['day']
f7 = f7.reset_index(level=[0,1]).drop(['match','date'],axis=1).rename(columns = {'level_0':'id'}

f8 = df['text'].str.extractall(r'(?P<year>[a-zA-Z][12]\d{3}[a-zA-Z]?)')
f8['year']= f8['year'].str.replace('[^0-9]','')
f8['year'] = f8['year'].apply(lambda x: x.strip())
f8['month'] = '01'
f8['day'] = '01'
f8['data8'] = f8['year']+'-'+f8['month']+'-'+f8['day']
```

```
f8 = f8.reset_index(level=[0,1]).drop(['match'],axis=1).rename(columns = {'level_0':'id'})

f9 = df['text'].str.extractall(r'(?P<date>(?P<month>\W?[a-zA-Z]?\d?\d)[-/](?P<day>\d?\d)[-/](?P<
f9['day'] = f9['day'].str.replace('[^0-9]','')
f9['month'] = f9['month'].str.replace('[^0-9]','')
f9['year'] = f9['year'].str.replace('[^0-9]','')
f9['day'] = f9['day'].apply(lambda x: x.strip())
f9['month'] = f9['month'].apply(lambda x: x.strip())
f9['year'] = f9['year'].apply(lambda x: x.strip())
f9['day'] = f9['day'].astype('int').map('{:02}'.format)
f9['month'] = f9['month'].astype('int').map('{:02}'.format)
f9['data9'] = f9['year']+'-'+f9['month']+'-'+f9['day']
f9 = f9.reset_index(level=[0,1]).drop(['match','date'],axis=1).rename(columns = {'level_0':'id'}

f10 = df['text'].str.extractall(r'(?P<date>(?P<month>\W?[a-zA-Z]?\d?\d)[-/](?P<day>\d?\d)[-/](?P
f10['day'] = f10['day'].str.replace('[^0-9]','')
f10['month'] = f10['month'].str.replace('[^0-9]','')
f10['year'] = f10['year'].str.replace('[^0-9]','')
f10['day'] = f10['day'].apply(lambda x: x.strip())
f10['month'] = f10['month'].apply(lambda x: x.strip())
f10['year'] = f10['year'].apply(lambda x: x.strip())
f10['day'] = f10['day'].astype('int').map('{:02}'.format)
f10['month'] = f10['month'].astype('int').map('{:02}'.format)
f10['data10'] = '19'+f10['year']+'-'+f10['month']+'-'+f10['day']
f10 = f10.reset_index(level=[0,1]).drop(['match','date'],axis=1).rename(columns = {'level_0':'id

f11 = df['text'].str.extractall(r'(?P<date>(?P<month>[^/]\d?\d)[-/](?P<year>[1-2]\d{3}\W))')
f11['month'] = f11['month'].str.replace('[^0-9]','')
f11['year'] = f11['year'].str.replace('[^0-9]','')
f11['month'] = f11['month'].apply(lambda x: x.strip())
f11['year'] = f11['year'].apply(lambda x: x.strip())
f11['month'] = f11['month'].astype('int').map('{:02}'.format)
f11['day'] = '01'
f11['data11'] = f11['year']+'-'+f11['month']+'-'+f11['day']
f11 = f11.reset_index(level=[0,1]).drop(['match','date'],axis=1).rename(columns = {'level_0':'id

f1.drop(['day','month','year'],axis=1,inplace=True)
f2.drop(['day','month','year'],axis=1,inplace=True)
f3.drop(['day','month','year'],axis=1,inplace=True)
f4.drop(['day','month','year'],axis=1,inplace=True)
f5.drop(['day','month','year'],axis=1,inplace=True)
f6.drop(['day','month','year'],axis=1,inplace=True)
f7.drop(['day','month','year'],axis=1,inplace=True)
f8.drop(['day','month','year'],axis=1,inplace=True)
f9.drop(['day','month','year'],axis=1,inplace=True)
f10.drop(['day','month','year'],axis=1,inplace=True)
f11.drop(['day','month','year'],axis=1,inplace=True)
```

```
data = f1.merge(f2,how='outer',left_on='id',right_on = 'id',
             sort=True).merge(f3,how='outer',left_on='id',right_on = 'id',
             sort=True).merge(f4,how='outer',left_on='id',right_on = 'id',
             sort=True).merge(f5,how='outer',left_on='id',right_on = 'id',
             sort=True).merge(f6,how='outer',left_on='id',right_on = 'id',
             sort=True).merge(f7,how='outer',left_on='id',right_on = 'id',
             sort=True).merge(f8,how='outer',left_on='id',right_on = 'id',
             sort=True).merge(f9,how='outer',left_on='id',right_on = 'id',
             sort=True).merge(f10,how='outer',left_on='id',right_on = 'id',
             sort=True).merge(f11,how='outer',left_on='id',right_on = 'id',
             sort=True).fillna(';')


data['data'] = data['data1']+'/'+data['data2']+'/'+data['data3']+'/'+data['data4']+'/'+data['dat

data.drop(['data1','data2','data3','data4','data5','data6','data7','data8','data9','data10','dat

data['data'] = data.data.str.replace('[^0-9-]',' ').apply(lambda x: x.strip())

data = data.join(data['data'].str.split(' ',2,expand=True)).rename(columns={0:'A',1:'B',2:'C'}).

data['id'] = data['id'].astype('int')

all_indeces = np.arange(0,500,1)
total_indeces= np.concatenate((np.array(data['id']),all_indeces), axis = 0)
(unique, counts) = np.unique(total_indeces,return_counts = True)
frequencies = np.asarray((unique, counts)).T
table=pd.DataFrame(frequencies, columns = ['index', 'count'])
missing_indeces = table[table['count']==1]
double_counts = table[table['count']==3]
missing_array = np.array(missing_indeces['index'])
df.iloc[missing_indeces['index'],:]

f13 = df.iloc[missing_indeces['index'],:]['text'].str.extractall(r'(?P<date>(?P<month>\d{1})/(?P
f13['month'] = f13['month'].str.replace('[^0-9]','')
f13['year'] = f13['year'].str.replace('[^0-9]','')
f13['month'] = f13['month'].apply(lambda x: x.strip())
f13['year'] = f13['year'].apply(lambda x: x.strip())
f13['month'] = f13['month'].astype('int').map('{:02}'.format)
f13['day'] = '01'
f13['data13'] = f13['year']+'-'+f13['month']+'-'+f13['day']
f13 = f13.reset_index(level=[0,1]).drop(['match','date','year','day','month'],axis=1).rename(col

f14 = df.iloc[missing_indeces['index'],:]['text'].str.extractall(r'(?P<date>(?P<year>[^/][1-2]\d
f14['year'] = f14['year'].str.replace('[^0-9]','')
f14['year'] = f14['year'].apply(lambda x: x.strip())
f14['day'] = '01'
```

```
f14['month'] = '01'
f14['data14'] = f14['year']+'-'+f14['month']+'-'+f14['day']
f14 = f14.reset_index(level=[0,1]).drop(['match','date','year','day','month'],axis=1).rename(col

month3 = {'Janaury':'01','Decemeber':'12'}

f15 = df.iloc[missing_indeces['index'],:]['text'].str.extractall(r'(?P<date>(?P<month>(Jan[a-z]+
f15['month'] = f15['month'].str.replace('[\W]','')
f15['year'] = f15['year'].str.replace('[\W]','')
f15['month'] = f15['month'].apply(lambda x: x.strip())
f15['year'] = f15['year'].apply(lambda x: x.strip())
f15['month'] = f15['month'].map(month3)
f15['day'] = '01'
f15['data15'] = f15['year']+'-'+f15['month']+'-'+f15['day']
f15 = f15.reset_index(level=[0,1]).drop(['match','date',2,'month','year','day'],axis=1).rename(c

data_missing = f15.merge(f13,how='outer',left_on='id',right_on = 'id',
            sort=True).merge(f14,how='outer',left_on='id',right_on = 'id',
            sort=True).fillna(';')


data_missing['data_missing'] = data_missing['data13']+'/'+data_missing['data14']+'/'+data_missin

data_missing.drop(['data13','data14','data15'],axis=1,inplace=True)

data_missing['data_missing'] = data_missing.data_missing.str.replace('[^0-9-]',' ').apply(lambda

data_missing = data_missing.join(data_missing['data_missing'].str.split(' ',1,expand=True)).rena

data_missing.iloc[1,2] = data_missing.iloc[1,3]

data_missing.drop(['data_missing', 'E'],axis=1,inplace=True)

data2 = data.merge(data_missing, how = 'outer', left_on='id',right_on='id',sort=True).fillna(';'

data2['A'] = data2['A']+'/'+data2['D']

data2.drop(['D'],axis=1,inplace=True)

data2['A'] = data2.A.str.replace('[^0-9-]',' ').apply(lambda x: x.strip())

all_indeces = np.arange(0,500,1)
total_indeces= np.concatenate((np.array(data2['id']),all_indeces), axis = 0)
(unique, counts) = np.unique(total_indeces,return_counts = True)
frequencies = np.asarray((unique, counts)).T
table=pd.DataFrame(frequencies, columns = ['index', 'count'])
missing_indeces = table[table['count']==1]
double_counts = table[table['count']==3]
```

```
missing_array = np.array(missing_indeces['index'])
df.iloc[missing_indeces['index'],:]

f16 = df.iloc[missing_indeces['index'],:]['text'].str.extractall(r'(?P<date>(?P<year>[1-2]\d{3}\
f16['year'] = f16['year'].str.replace('[^0-9]','')
f16['year'] = f16['year'].apply(lambda x: x.strip())
f16['day'] = '01'
f16['month'] = '01'
f16['data16'] = f16['year']+'-'+f16['month']+'-'+f16['day']
f16 = f16.reset_index(level=[0,1]).drop(['match','date','year','day','month'],axis=1).rename(col

data3 = data2.merge(f16, how = 'outer', left_on='id',right_on='id',sort=True).fillna(';')

data3['A'] = data3['A']+'/'+data3['data16']

data3.drop(['data16'],axis=1,inplace=True)

data3['A'] = data3.A.str.replace('[^0-9-]',' ').apply(lambda x: x.strip())

data3[data3.index != data3['id']]

data3.iloc[70:80,:]

data3.drop(73, axis=0, inplace = True)

data3 = data3.sort_values('A', ascending = True).reset_index().drop('index', axis=1)

return pd.Series(data3['id'])
```

date_sorter()
def date_sorter(): import re

```
year = df['text'].str.extractall(r'(?P<year>.?[12]\d{3}.?)')
year['year']= year['year'].str.replace('[^0-9]','')
year['year'] = year['year'].apply(lambda x: x.strip())
year = year.reset_index(level=1).drop('match',axis=1)

day_month1 = df['text'].str.extractall(r'(?P<day_month1>(?P<day>\W\d?\d)\W(?P<month>\W?Jan\W?|\W
day_month1['day'] = day_month1['day'].str.replace('[.,;:/\)(-]','')
day_month1['month'] = day_month1['month'].str.replace('[.,;:/\)(-]','')
day_month1['day'] = day_month1['day'].apply(lambda x: x.strip())
day_month1['month'] = day_month1['month'].apply(lambda x: x.strip())
day_month1 = day_month1.reset_index(level=1).drop('match',axis=1)

day_month2 = df['text'].str.extractall(r'(?P<day_month2>(?P<day>\W\d?\d)\W(?P<month>\W?January\W
day_month2['day'] = day_month2['day'].str.replace('[.,;:/\)(-]','')
day_month2['month'] = day_month2['month'].str.replace('[.,;:/\)(-]','')
day_month2['day'] = day_month2['day'].apply(lambda x: x.strip())
```

```
day_month2['month'] = day_month2['month'].apply(lambda x: x.strip())
day_month2 = day_month2.reset_index(level=1).drop('match',axis=1)

month_day1 = df['text'].str.extractall(r'(?P<month_day1>(?P<month>\W?Jan\W?|\W?Feb\W?|\W?Mar\W?|
month_day1['day'] = month_day1['day'].str.replace('[.,;:/\)(-]','')
month_day1['month'] = month_day1['month'].str.replace('[.,;:/\)(-]','')
month_day1['day'] = month_day1['day'].apply(lambda x: x.strip())
month_day1['month'] = month_day1['month'].apply(lambda x: x.strip())
month_day1 = month_day1.reset_index(level=1).drop('match',axis=1)

month_day2 =df['text'].str.extractall(r'(?P<month_day2>(?P<month>\W?January\W?|\W?February\W?|\W
month_day2['day'] = month_day2['day'].str.replace('[.,;:/\)(-]','')
month_day2['month'] = month_day2['month'].str.replace('[.,;:/\)(-]','')
month_day2['day'] = month_day2['day'].apply(lambda x: x.strip())
month_day2['month'] = month_day2['month'].apply(lambda x: x.strip())
month_day2 = month_day2.reset_index(level=1).drop('match',axis=1)

day_month_year1 = df['text'].str.extractall(r'(?P<day_month_year>(?P<month>\d?\d)[-/]{1}(?P<day>
day_month_year1['day'] = day_month_year1['day'].str.replace('[^0-9]','')
day_month_year1['month'] = day_month_year1['month'].str.replace('[^0-9]','')
day_month_year1['year'] = day_month_year1['year'].str.replace('[^0-9]','')
day_month_year1['day'] = day_month_year1['day'].apply(lambda x: x.strip())
day_month_year1['month'] = day_month_year1['month'].apply(lambda x: x.strip())
day_month_year1['year'] = day_month_year1['year'].apply(lambda x: x.strip())
day_month_year1 = day_month_year1.reset_index(level=1).drop('match',axis=1)

day_month_year2 = df['text'].str.extractall(r'(?P<day_month_year>(?P<month>\d?\d)[-/]{1}(?P<day>
day_month_year2['day'] = day_month_year2['day'].str.replace('[^0-9]','')
day_month_year2['month'] = day_month_year2['month'].str.replace('[^0-9]','')
day_month_year2['year'] = day_month_year2['year'].str.replace('[^0-9]','')
day_month_year2['day'] = day_month_year2['day'].apply(lambda x: x.strip())
day_month_year2['month'] = day_month_year2['month'].apply(lambda x: x.strip())
day_month_year2['year'] = day_month_year2['year'].apply(lambda x: x.strip())
day_month_year2 = day_month_year2.reset_index(level=1).drop('match',axis=1)

month_year = df['text'].str.extractall(r'(?P<month_year>(?P<month>[^\d/]\d?\d)/(?P<year>[12]\d{3
month_year['month'] = month_year['month'].str.replace('[^0-9]','')
month_year['year'] = month_year['year'].str.replace('[^0-9]','')
month_year['month'] = month_year['month'].apply(lambda x: x.strip())
month_year['year'] = month_year['year'].apply(lambda x: x.strip())
month_year = month_year.reset_index(level=1).drop('match',axis=1)

month1 = {'Jan':'01','Feb':'02','Mar':'03','Apr':'04','May':'05','Jun':'06',
            'Jul':'07','Aug':'08','Sep':'09','Oct':'10','Nov':'11','Dec':'12'}

month2 = {'January':'01','February':'02','March':'03','April':'04','May':'05','June':'06',
            'July':'07','August':'08','September':'09','October':'10','November':'11','December
```

```
date1 = day_month1.merge(year, how ='inner', left_index = True, right_index = True)
date2 = day_month2.merge(year, how ='inner', left_index = True, right_index = True)
date3 = month_day1.merge(year, how ='inner', left_index = True, right_index = True)
date4 = month_day2.merge(year, how ='inner', left_index = True, right_index = True)
month_year['day'] = '01'
year['day'] = '01'
year['month'] = '01'

date1['month'] = date1['month'].map(month1)
date2['month'] = date2['month'].map(month2)
date3['month'] = date3['month'].map(month1)
date4['month'] = date4['month'].map(month2)

date1 = date1.astype('int')
date2 = date2.astype('int')
date3 = date3.astype('int')
date4 = date4.astype('int')
month_year = month_year.astype('int')
day_month_year1 = day_month_year1.astype('int')
day_month_year2 = day_month_year2.astype('int')
year = year.astype('int')

date1["month"] = date1.month.map("{:02}".format)
date2["month"] = date2.month.map("{:02}".format)
date3["month"] = date3.month.map("{:02}".format)
date4["month"] = date4.month.map("{:02}".format)
month_year["month"] = month_year.month.map("{:02}".format)
day_month_year1["month"] = day_month_year1.month.map("{:02}".format)
day_month_year2["month"] = day_month_year2.month.map("{:02}".format)
year['month'] = year.month.map("{:02}".format)

date1["day"] = date1.day.map("{:02}".format)
date2["day"] = date2.day.map("{:02}".format)
date3["day"] = date3.day.map("{:02}".format)
date4["day"] = date4.day.map("{:02}".format)
day_month_year1["day"] = day_month_year1.day.map("{:02}".format)
day_month_year2["day"] = day_month_year2.day.map("{:02}".format)
year['day'] = year.day.map("{:02}".format)
month_year["day"] = month_year.day.map("{:02}".format)

date1 = date1.astype('str')
date2 = date2.astype('str')
date3 = date3.astype('str')
date4 = date4.astype('str')
month_year = month_year.astype('str')
day_month_year1 = day_month_year1.astype('str')
day_month_year2 = day_month_year2.astype('str')
year = year.astype('str')
```

```
date1['date'] = date1['year']+'/'+date1['month']+'/'+date1['day']
date2['date'] = date2['year']+'/'+date2['month']+'/'+date2['day']
date3['date'] = date3['year']+'/'+date3['month']+'/'+date3['day']
date4['date'] = date4['year']+'/'+date4['month']+'/'+date4['day']
month_year['date'] = month_year['year']+'/'+month_year['month']+'/'+month_year['day']
day_month_year1['date'] = day_month_year1['year']+'/'+day_month_year1['month']+'/'+day_month_yea
day_month_year2['date'] = '19'+day_month_year2['year']+'/'+day_month_year2['month']+'/'+day_mont
year['date'] = year['year']+'/'+year['month']+'/'+year['day']

date1 = date1.drop(['month', 'day','year'], axis=1).reset_index().rename(columns={'date':'date1'
date2 = date2.drop(['month', 'day','year'], axis=1).reset_index().rename(columns={'date':'date2'
date3 = date3.drop(['month', 'day','year'], axis=1).reset_index().rename(columns={'date':'date3'
date4 = date4.drop(['month', 'day','year'], axis=1).reset_index().rename(columns={'date':'date4'
month_year = month_year.drop(['month', 'day','year'], axis=1).reset_index().rename(columns={'dat
day_month_year1 = day_month_year1.drop(['month', 'day','year'], axis=1).reset_index().rename(col
day_month_year2 = day_month_year2.drop(['month', 'day','year'], axis=1).reset_index().rename(col
year = year.drop(['month', 'day','year'], axis=1).reset_index().rename(columns={'date':'date8'})

dates =  date1.merge(date2,how='outer',left_on='index',right_on='index', sort = True
                    ).merge(date3,how='outer',left_on='index',right_on='index', sort =True
                           ).merge(date4,how='outer',left_on='index',right_on='index', sort =Tru
                                  ).merge(month_year,how='outer',left_on='index',right_on='index
                                         ).merge(day_month_year1,how='outer',left_on='index',rig
                                                ).merge(day_month_year2,how='outer',left_on='ind
                                                       ).merge(year, how = 'outer', left_on='ind
                                                              )




dates = dates.fillna('a').astype('str')

dates['date1'] = dates['date1']+';'+dates['date2']+';'+dates['date3']+';'+dates['date4']+';'+dat

dates.drop(['date2','date3','date4','date5','date6','date7','date8'],axis=1, inplace = True)

dates['date1'] = dates['date1'].str.replace('[a;]',' ')

dates['date1'] = dates['date1'].str.strip()




dates['index'] = dates['index'].astype('int')
```

```
#all_indeces = np.arange(0,500,1)
#total= np.concatenate((np.array(dates['index']),all_indeces), axis = 0)
#(unique, counts) = np.unique(total,return_counts = True)
#frequencies = np.asarray((unique, counts)).T
#tableau=pd.DataFrame(frequencies, columns = ['index', 'count'])
#missing = tableau[tableau['count']==1]
#double_count = tableau[tableau['count']==3]
#missing_array = np.array(missing['index'])
#df.iloc[missing['index'],:]
#df.iloc[double_count['index'],:]

dates[dates.index != dates['index']]
dates.iloc[70:80,:]

dates.drop(73, axis=0, inplace = True)

data = dates.join(dates['date1'].str.split(' ', 2, expand=True).rename(columns={0:'A', 1:'B',2:'



return pd.Series(data['index'])
```

date_sorter()