13. エクセルのデータを作図する

プログラミング・データサイエンス I

2021/7/15

1　今日の目的

今日の目的

- Excel ファイルのデータを作図する
- `DataFrame` や `Series` の作図メソッド
- 九州と佐賀の人口を例に

前回は、プログラムの中に、作図するデータがありました。今日は、エクセルにあるデータを読み込んで、作図しましょう。サンプルプログラムを取得してください。

`https://github.com/first-programming-saga/SagaPopulation`

2　Excel データの作図: Drawing with data in Excel

Excel データの作図: Drawing with data in Excel

- `DataFrame` から `Series` を取り出す
- `Series` のメソッドで作図する

以前に、エクセルファイルを扱うライブラリ `Pandas` を用いました。エクセルファイルを読みこむと `DataFrame` という表形式のデータになることを説明しました。この `Pandas` は、前回利用した `matplotlib` と連携した作図のための機能を持っています。つまり、`matplotlib` がデータ列として利用していたリストを変形することなく、`Pandas` のデータのままで作図することができます。

プログラムと一緒に、`KyushuPopulation.xlsx` という Excel ファイルがあります。表 1 のように、国勢調査に基づく九州各県の人口推移が 5 年毎に入っています。一行取り出すと、一つの県の人口の時系列になります。はじめに、これをプロットしましょう。

表 1　九州各県の人口推移

県	1990	1995	2000	2005	2010	2015
福岡	4,758	4,873	4,968	5,014	5,039	5,120
佐賀	882	885	884	874	859	847
長崎	1,573	1,554	1,532	1,502	1,450	1,413
熊本	1,848	1,862	1,870	1,858	1,834	1,818
大分	1,246	1,243	1,236	1,225	1,207	1,191
宮崎	1,182	1,187	1,187	1,173	1,153	1,136
鹿児島	1,805	1,794	1,787	1,763	1,722	1,691

ソースコード 2.1　Series をプロットする

```
1 def drawTimeSequence(df): #dataFrame を受け取る
2     for pref in df.index: #各行にある県名を取得
3         ser = df.loc[pref] #一行のデータ（時系列）を取得
4         #一行のデータをプロットし、ラベルとして県名を設定
5         ser.plot(label = pref, color = colors[pref])
```

KyushuPopulation.ipynb を開けてください。データを描画している部分が、ソースコード 2.1 です。DataFrame である df を引数として受け取り、各行、つまり各県のデータを Series である ser として取り出します。最後の行で、各県のデータをプロットしています。

ソースコード 2.2　Series をプロットする

```
1  filename = 'KyushuPopulation.xlsx'
2  with pandas.ExcelFile(filename) as f:
3      df = pandas.read_excel(f)
4  df.set_index('県', inplace = True) #左端のカラムにインデクスを設定
5  #totalPopulation(df)
6  #図の準備
7  plt.figure(figsize = (15, 15))
8  plt.rcParams['font.size'] = 20
9  plt.title('九州の人口')
10 plt.xlabel('年')
11 plt.ylabel('人口 ($\\times1,000$)')
12 #各県のデータをプロット
13 drawTimeSequence(df)
14
15 plt.legend()
16 plt.show()
```

Series 形式で、各県のデータを取り出したため、年と組になったデータとなります。そのため、年を横軸として折れ線を引くことができます。

課題 1　ソースコード 2.1 の 5 行目はコメントアウトしています。ここで呼び出しているのは、ソースコード 2.3 の部分で、九州全体の人口の総和を求め、df に追加する部分です。コメントアウトを外して実行し、動作を確かめなさい。

ソースコード 2.3　九州全体の人口を計算する

```
1  def totalPopulation(df):
2      total = df.sum(axis = 'index')
3      df.loc['九州全体'] = total
```

次は、九州各県の人口の割合を図示しましょう。このような場合には、円グラフを使うことが多いでしょう。KyushuPopulationPie.ipynb を開けてください。

ソースコード 2.4　1995 年と 2015 年の人口比率

```
1  #図の準備
2  year = [1995, 2015]
3  size = 10
4  xsize = size * len(year)
5  fig, ax = plt.subplots(1, len(year), figsize=(xsize, size))
6  plt.rcParams['font.size'] = 20
7  plt.suptitle('九州の人口')
8  #各県のデータをプロット
9  for i in range(len(year)):
10     ser = df[year[i]]
11     ser.name = '' #Series の名前を消去：消さないと左に小さく表示されてしまう
12     ser.plot.pie(ax = ax[i], startangle = 90, counterclock = False,
13         autopct = '%1.2f%%', title = str(year[i]))
14 plt.show()
```

ソースコード 2.4 を見てください。1995 年と 2015 年とで、九州全体の人口に対する各県の割合の変化を円グラフにするコードです。一行二列の図を ax に描くことにします。6 行目は、全体に対するタイトルを指定しています。8 行目からの for ループで、各年の図を作成しています。実行して、確認してください。

課題 2　1995 年、2005 年、2015 年の三つを作図するように、ソースコード 2.4 を変更し、動作を確かめなさい。

3　オンラインデータを扱う: 佐賀県の人口

実データを扱う: 佐賀県の人口

- 「ビックデータ&オープンデータ・イニシアティブ九州」のデータを活用
  `https://www.bodik.jp/`
- 人口の推移を図示

インターネット上には、様々なデータが公開されています。佐賀県が公開している人口推移のデータを使って作図をしましょう。配布したファイル中にある `jinkou.xlsx` を開けてください。このファイルは、以下の URL にあります。

`http://data.bodik.jp/dataset/77e0cc66-c15d-4473-b3df-2664fe8e2e63/resource/8dc71515-526a-4168-866c-05d2cc8dad7b/download/jinkou.xlsx`

A 列と B 列に、各行のラベルが記載されています。二つのセルが連結しているものもあります。そのため、行のラベルに使える列を指定することは難しい状況です。一方、V 列にはラベルに使える文字列が格納されています。こちらを行のラベルに使いましょう。

4 行目が調査年に対応する列のラベルです。しかし、和暦であり、更に元号が省略されています。西暦との対応表が必要になります。

また、データが入っていない行があるとともに、31 行目以降は説明が入っています。除いておきましょう。

それでは、`population.ipynb` を開いてください。`jinkou.xlsx` は閉じてください。

ソースコード 3.1　Excel の読み込み

```
1 url = 'http://data.bodik.jp/dataset/77e0cc66-c15d-4473-b3df-2664fe8e2e63/
      resource/8dc71515-526a-4168-866c-05d2cc8dad7b/download/jinkou.xlsx'
2 # url = 'jinkou.xlsx'
3 #Excel の内容を DataFrame へ
4 data = pandas.read_excel(url, header = 3, usecols = 'C:V',
5     index_col = 19, skiprows = [4, 18, 27], skipfooter = 8)
```

ソースコード 3.1 は、`jinkou.xlsx` を読みこむ部分です。インターネットからの読み込みがうまく行かない場合には、ローカル側の `jinkou.xlsx` を使うように、コメントアウトを変更してください。

前述した問題のうち、調査年の問題以外を解決しています。引数は、以下のような目的で使用しています。

1. ファイル名
2. 列の名前として使用する行
3. 使用する列の範囲
4. 行の名前として使用する列 (列の範囲を制限したことに注意)
5. 使用しない行
6. 末尾にある説明行を無視

データの中には、値が入っておらず、代わりに”-”が入っている部分があります。ここは、そのままにしておきます。

調査年の問題は、今回は対応表を作ることで解決します。ソースコード 3.2 を見てください。列の名前に対して、西暦を対応させる辞書型のデータです。

ソースコード 3.2　和暦から西暦へ

```
#エクセル中の和暦ラベルと西暦の対応付け
wareki={'大正 9年':1920, '昭和 5年':1930, '10年':1935, '15年':1940,
        '20年':1945, '25年':1950, '30年':1955, '35年':1960,'40年':1965,
        '45年':1970, '50年':1975, '55年':1980, '60年':1985, '平成 2年':1990,
        '7年':1995, '12年':2000, '17年':2005, '22年':2010, '27年':2015}
```

ソースコード 3.3　指定した行の作図

```
def plotSub(plotList, df):
    for loc, label in plotList:
        ser = getSeries(df, loc)
        ser.plot(label = label, linewidth = 3)

def getSeries(data,row):
    vList, yearList = [], []
    for k in wareki.keys():
        population = data[k][row] / 1000.
        year = wareki[k] #和暦から西暦へ
        vList.append(population)
        yearList.append(year)
    return pandas.Series(vList, index = yearList)
```

ソースコード 3.3 が作図の中核部分です。関数 `plotSub()` の第一引数は、シート内の行のラベルと作図の際に使うラベルの組の tuple のリストです。第二引数はシートに対応する `DataFrame` です。リスト `plotList` の要素は、二つの要素を持つ tuple ですから、それぞれを `loc`、`label` として取り出します。3 行目で `getSeries()` を使って、指定した行に対応する `Series` を取得し、4 行目で折れ線で作図します。

関数 `getSeries()` では、初めに二つの空のリスト `vList` と `yearList` を作成します。調査年に相当するセルの値を `population` として取り出し、`vList` に追加します。一方で、和暦に対応する西暦を `yearList` に追加します。最後に、`vList` の各値に、`yearList` の各値をインデクスとして対応付けた `Series` を返します。

実行結果を図 1 に示します。

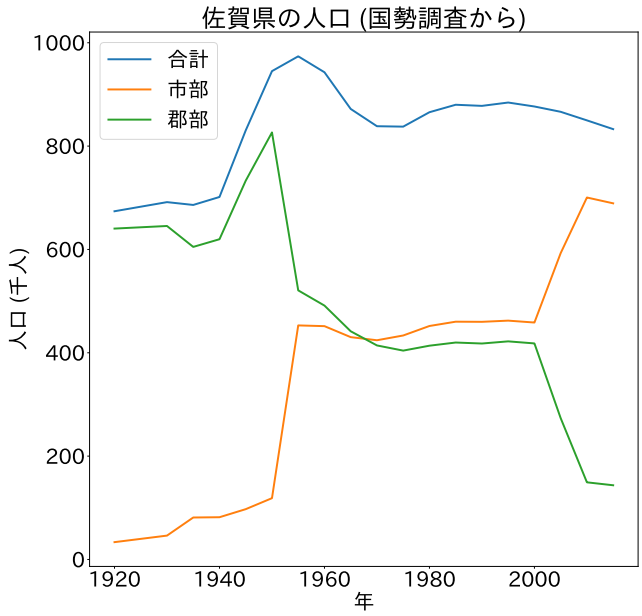

図 1　佐賀県の人口推移：市部と郡部

課題 3　シートには、年代層別の人口がありました。年少 (0〜14)、生産年齢 (15〜64)、老年 (65 以上) の人口の推移を作図しなさい。