11. 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

2023/6/29

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、作図対象のデータがありました。しかし、多くの場合にはプログラムとは別にデータファイルがあります。今日は、エクセルファイルに保存されているデータを読み込んで、作図しましょう。

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。

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` のメソッドを使って作図する

2.1 九州各県の人口推移

プログラムと一緒に、KyushuPopulation.xlsx という Excel ファイルを配布しています。このファイルには、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 年毎のデータとして入っています。県名の行取り出すと、その県の人口の時系列になります。はじめに、これを可視化しましょう。

表 1　九州各県の人口推移 (千人)

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

KyushuPopulation.ipynb を開けてください。各県のデータを描画している部分が、ソースコード 2.1 です。この関数 drawTimeSequence() は、DataFrame である df を引数として受け取ります。その DataFrame の各行、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、Series のメソッドである plot() を使って各県の時系列データをプロットしています。ここで、plot() メソッドの引数について整理しておきます。

ax　　作図先の Axes を指定
label　線に対するラベル
color　線の色

Series 形式で、各県のデータを取り出したため、単なる数値のリストではなく、元の DataFrame の列のラベルである年と組になったデータとなります。そのため、plot() メソッドで作図すると、年を横軸として折れ線を引くことができます。

課題 2.1　Series 形式になったデータを print() を使って、その内容を確認しなさい。

ソースコード 2.1　各県の時系列 Series をプロットする

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

ソースコード 2.2　各県人口の時系列をプロットする

```
def drawData(df: pandas.DataFrame):
    #図の準備
    fig, ax = plt.subplots(figsize = (10, 10), facecolor = 'white')
    ax.set_title('九州の人口')
    ax.set_xlabel('年')
    ax.set_ylabel('人口 ($\\times1,000$)')
    #各県のデータをプロット
    drawTimeSequence(df, ax)

    plt.legend()
    plt.show()
```

ソースコード 2.3　各県人口の時系列をプロットするメイン部分

```
filename = 'KyushuPopulation.xlsx'
with pandas.ExcelFile(filename) as f:
    df = pandas.read_excel(f)
df.set_index('県', inplace = True) #左端のカラムにインデクスを設定
#totalPopulation(df)
drawData(df)
```

課題 2.2　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 2.4 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 2.4　九州全体の人口を計算する

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

2.2　九州各県の人口割合

次に、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。円グラフを表示する関数も pandas に用意されています。KyushuPopulationPie.ipynb を開けてください。

ソースコード 2.5　1995 年と 2015 年の人口比率

```
def drawPi(df:pandas.DataFrame,year:list[int],size:int):
    xsize = size * len(year)
    fig, ax = plt.subplots(1, len(year), figsize=(xsize, size),
            facecolor = 'w')
    plt.suptitle('九州の人口')
    #各県のデータをプロット
    for i,y in enumerate(year):
        ser = df[y]
        #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
        ser.name = ''
        ser.plot.pie(ax = ax[i], startangle = 90,
            counterclock = False, autopct = '%1.2f%%', title = str(y))

    plt.savefig('KyushuPopulationPie.pdf')
    plt.show()
```

ソースコード 2.5 を見てください。リストには、作図対象となる年が整数で入っています。大きさを N とします。それらの対象年ついて、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 N 列の図を描くので、subplots() を使って ax をつくり、そこに図を描くことにします。

5 行目は、全体に対するタイトルを指定しています。7 行目からの for ループで、各年の図を作成しています。対象となる年を指定することで、DataFrame から列を Series として取り出し、作図しています。11 行目の plot.pie() の最初の引数に、作図先となる ax を指定しています。結果を図 1 に示します。実行して、確認してください。実行結

果を見ると、自動で百分率を計算し、表示していることがわかります。plot.pie() の引数 autopct で指定しているのは、百分率表示の桁数です。

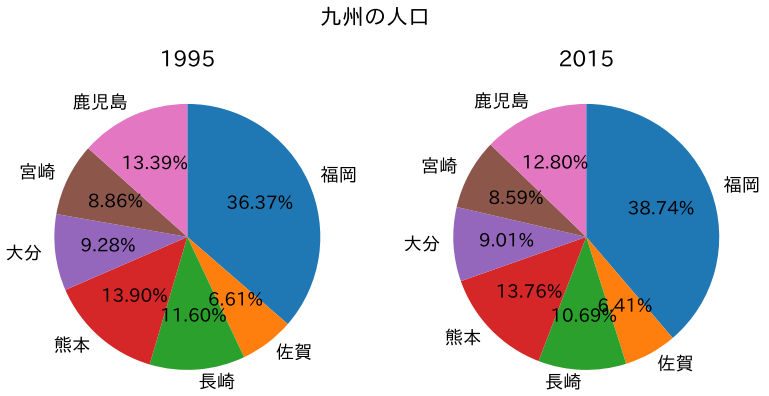

図 1　九州各県の人口

課題 2.3　1995 年、2005 年、2015 年の三つを作図するように、ソースコード 2.5 を変更し、動作を確かめなさい。

3　実データを扱う: 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

以前に見たものと同じファイルです。列のラベルの西暦化、V 列を行のラベルに使う、空白行を除くなどの処理を、以前と同じように行っておきます。

作図プログラムは population.ipynb です。ソースコード 3.1 が作図の中核部分です。

ソースコード 3.1　指定した行の作図

```
plotList=['佐賀県', '市部','郡部']
for label in plotList:
    ser = data.loc[label]/1000
    ser.plot(ax = ax, label = label, linewidth = 3)
```

plotList に行のラベルを指定しています。各行について、そのデータを 1/1000 しているのが、2 行目です。このように、Series の各要素に対して、一括して処理を行うことができます。3 行目で、データを折れ線で作図します。実行結果を図 2 に示します。

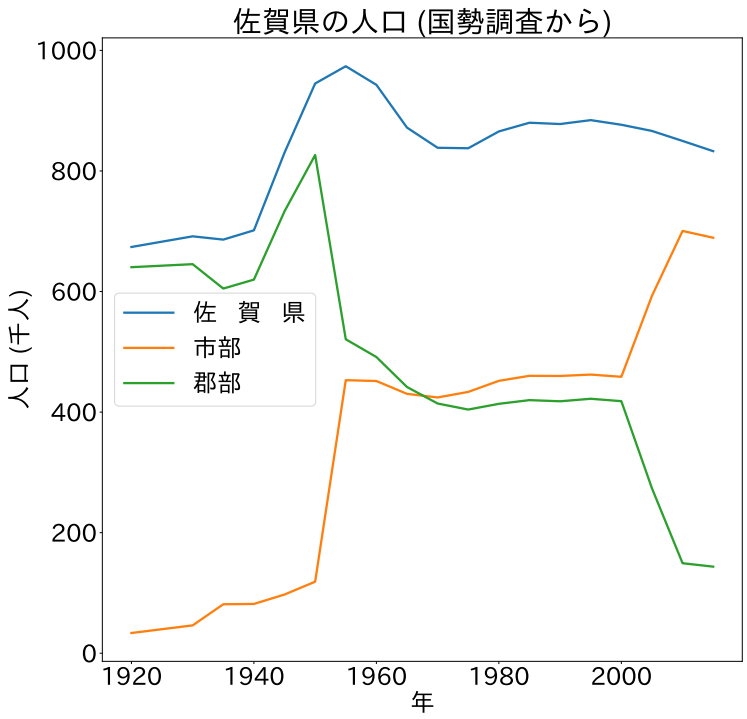

図 2　佐賀県の人口推移: 市部と郡部

課題 3.1　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。データ中の行の名前を確認する必要があります。

4　積み上げグラフ:Stacked Bar Charts

各年齢層の占める割合の変化を示す方法として、前節の折れ線グラフのほかに、積み上げグラフがあります (図 3)。

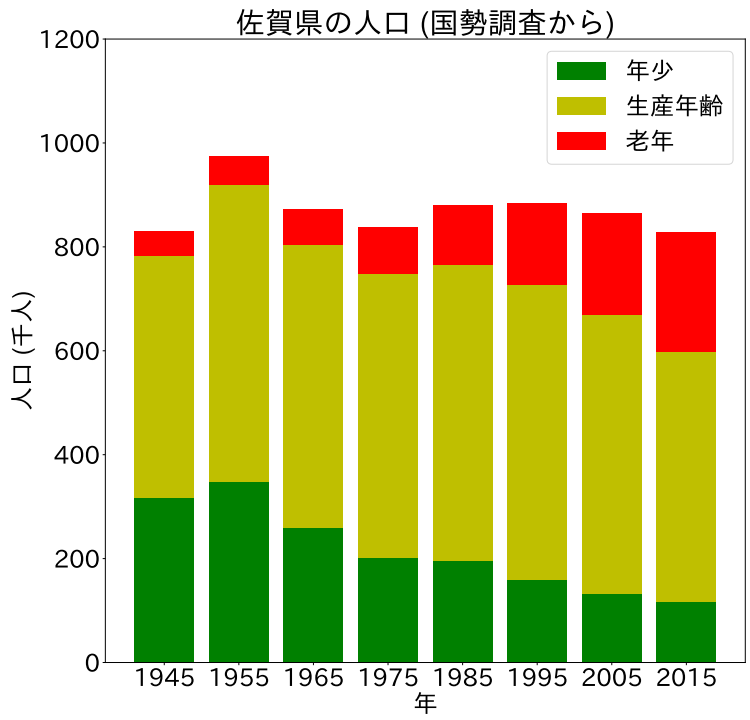

図 3　佐賀県の人口構成推移

`matplotlib` では、棒グラフを描く際に、その基点、つまり棒の縦方向の原点を指定するパラメタ `bottom` を持っています。これを使って作図しましょう。サンプルプログラムは `populationStackedBar.ipynb` です。

Excel を読み込んで成形した `DataFrame` 内には、年齢層を表すインデクスがあります。これを使うと、`Series` として取り出すことができるのは、前節で行いました。全部の列、つまり年を作図すると、数が多すぎますから、一部の年だけをを取り出すことにします。`Series.filter()` というメソッドは、引数に `Series` 中のインデクスのリストを渡すと、対応する部分だけからなる `Series` を返します。つまり、引数に表示する年のリストを渡します。こうしてできる部分的データを作図すればよいでしょう。

ソースコード 4.1　積み上げグラフ

```
1  def plotSub(yearList:list[int], df:pandas.DataFrame, ax:plt.Axes):
2      """
3      yearList で指定された年の棒グラフを描画
4      """
5      #必要なデータに成形する：対象となる年のデータを取得し、1/1000 にする
6      population = [
7          df.loc['0〜14 歳'].filter(yearList)/1000,
8          df.loc['15〜64 歳'].filter(yearList)/1000,
9          df.loc['65 歳以上'].filter(yearList)/1000
10     ]
11     #積み上げの基点となる Series
12     bottom = pandas.Series([0]*len(yearList), index = yearList)
13     color = ['g', 'y', 'r']
14     label = ['年少', '生産年齢', '老年']
15     for k, p in enumerate(population):
16         p.plot.bar(ax = ax, bottom = bottom,
17             tick_label = yearList, color = color[k], label = label[k])
18         bottom += p
```

次は、棒グラフの基点です。作図する Series と同じインデクスを持つ Series となる bottom を用意しましょう。データを作図する度に、bottom にデータを加算します。こうすることで、bottom の上に、積み上げた棒グラフを描くことができます。

ソースコード 4.1 を見てください。引数 yearList には、作図対象の年が入っています。6 行目から 10 行目で、各年齢層に対する行のデータから、対象年を抜き出し、さらに 1/1000 に値を変換しています。

12 行目は、yearList を使って、すべての値が 0 である bottom という Series を作っています。0 の数は yearList の長さです。またそのインデクスは、yearList になっていることに注意してください。この段階では、bottom には、全て 0 が入っています。

16 行目は、加工した Series の棒グラフとしての作図です。bottom が基点となっています。作図した Series を 18 行目で bottom に加算していきます。例えば、最初に「年少」の人口を棒グラフにした後、bottom にはその値が加算され、次の「生産年齢」の棒グラフの基点となります。

課題 4.1　棒グラフの基点を表す bottom の値が変化する様子を確かめなさい。

課題 4.2　quiz.ipynb にある課題です。このプログラムは、KyushuPopulation.ipynb

を少し改変して、九州各県の人口割合の時系列を作図します。

5　次回

日本では、少子高齢化が大きな課題となっています。次回は、国立社会保障・人口問題研究所が作成した将来人口予測データを分析しましょう。