

9. Excel ファイルを作る: Creating Excel files

プログラミング・データサイエンス I

2022/6/16

1 今日の目的

今日の目的

- DataFrame データを作る
- Excel ファイルへ書き込む
- DataFrame データ内で平均等を計算し書き込む

前回は、Excel ファイルや CSV ファイルを読み込み、DataFrame という形式になることを学びました。列の名前の一覧、行の名前の一覧がありました。また、一列、一行のデータが Series という形式になることも学びました。

今日の最初のテーマは、Python の中で DataFrame 形式のデータを作成し、Excel ファイルへと書き出すことです。最初に、各列のデータから DataFrame 形式のデータを作ります。一旦、DataFrame 形式のデータが出来た後は、行や列を追加することができます。

Excel ファイルを使って、列の平均値を、一番下の行に付け加えるのは、よく行う操作です。応用問題として、これと同様の操作を Python で行いましょう。

それでは、今日のサンプルプログラムを取得してください。

<https://github.com/first-programming-saga/excelAndCSV2>

2 DataFrame データを作る

DataFrame データを作る

- 列のデータを準備して、行の名前を付ける
- 列のデータと、行の名前を準備して、作る

2.1 列とデータを指定する方法

最初の例 `createDataFrame.ipynb` を開きましょう。最初の作り方をソースコード 2.1 に示します。 `data` は、辞書型のデータです。キーとなっているのは科目名、値は点数のリストです。リストの長さは全て同じでなければならないことに注意が必要です。 `index` は、行のラベルのリストです。点数のリストと同じ長さでなければなりません。 `DataFrame` のコンストラクタに、 `data` と `list` を与えます。 `DataFrame` のコンストラクタとは、 `DataFrame` というデータ構造をもったオブジェクトを生成するメソッドです。最後に、一列目に「名前」という名前を付けています。生成した `DataFrame` データが出力例 2.1 です。

ソースコード 2.1 `DataFrame` データを作る

```
1 data = {'数学':[90, 70, 100, 60],
2         '理科':[80, 90, 90, 70],
3         '英語':[80, 100, 80, 90],
4         '社会':[70, 90, 70, 80]}
5 index = ['山田', '佐藤', '鈴木', '渡辺']
6 df = pandas.DataFrame(data, index)
7 df.index.name = '名前'
8 print(df)
```

	数学	理科	英語	社会
名前				
山田	90	80	80	70
佐藤	70	90	100	90
鈴木	100	90	80	70
渡辺	60	70	90	80

出力例 2.1: 生成したデータ

2.2 行の名前も指定する方法

`DataFrame` の作り方の別の方法をソースコード 2.2 に示します。 `data2` は、辞書型のデータです。前の例との違いは、名前リスト、つまり行の名前も `data2` に含んでいることです。 `DataFrame` を生成した直後で、「名前」という列をインデクスに指定しています。結果が、出力例 2.1 と同じであることを確認してください。

ソースコード 2.2 DataFrame データを作る

```
1 data2 = {'名前':['山田', '佐藤', '鈴木', '渡辺'],
2         '数学':[90, 70, 100, 60],
3         '理科':[80, 90, 90, 70],
4         '英語':[80, 100, 80, 90],
5         '社会':[70, 90, 70, 80]}
6 df2 = pandas.DataFrame(data2)
7 df2.set_index('名前', inplace = True)
8 print(df2)
```

3 行と列の追加

DataFrame データを変更する

- 列データを追加する
- 行データを追加する
- 空の行・列データを追加する
- Excel として出力する

3.1 リストを使う方法

次に新しいデータを追加しましょう。最初は、新しい列の追加です。ソースコード 3.1 を見てください。新しい科目名 label として「体育」を用意し、4 人分の成績のリスト record を準備します。作成した record を DataFrame データである df に追加しています。結果表示を確認してください。表示がずれているかもしれませんが、5 科目分の成績が登録できていることがわかります。

ソースコード 3.1 列データを追加する

```
1 label = '体育'
2 record = [90, 70, 80, 100]
3 df[label] = record
4 print(df2)
```

次は、ここに新しい行を追加します。ソースコード 3.2 です。新しい生徒「古賀」を登

録します。行の名前 `name` とそれに対応した一列のデータ `recordKoga` を準備します。5 科目分のデータが必要であることを注意します。このデータを `df` に追加しています。結果を確認してください。

ソースコード 3.2 行データを追加する

```
1 name = '古賀'  
2 recordKoga = [90, 90, 85, 70, 70]  
3 df.loc[name] = recordKoga
```

3.2 Series を使う方法

`DataFrame` から行や列を取り出すと `Series` という形式になりました。インデクスが付いた一次元のデータでした。この `Series` という形式のデータを生成して、行として追加する例を示します。

ソースコード 3.3 では、2 行目で `Series` という形式のデータを生成しています。引数 `index` には、`df` の列の名前の一覧 `df.columns` を指定するします。このようにすることで、最初の引数のデータリストに科目名を対応させています。

ソースコード 3.3 行データを追加する

```
1 name = '藤井'  
2 ser = pandas.Series([90, 75, 85, 100, 65], index = df.columns)  
3 df.loc[name] = ser
```

3.3 ファイルへの保存

最後に、Excel ファイルとして出力しましょう。出力もとても簡単です。ソースコード 3.4 では、`DataFrame` データをファイル `tmp-out.xlsx` へと出力しています。ソースコード 3.5 の形式も使えます。

ソースコード 3.4 Excel ファイルへと出力

```
1 df.to_excel('tmp-out.xlsx')
```

ソースコード 3.5 Excel ファイルへと出力

```
1 with pandas.ExcelWriter('tmp-out.xlsx') as f:  
2     df.to_excel(f)
```

4 課題：科目の平均を計算する

課題：科目の平均を計算する

- 行、列の間の演算
- 列方向の和
- 行方向の平均
- Excel ファイルを出力

実際に、成績表の Excel ファイルを読み込み、成績の処理をしてみましょう。StatFromExcel.ipynb を空けてください。ソースコード 4.1 が、その main 部分です。2 行目は、Excel を読み、一列目を行インデクスにしています。それ以降は、全てコメントアウトしています。順に動作を確認していきましょう。

ソースコード 4.1 StatFromExcel.ipynb の main 部分

```
1 filename = "data.xlsx"  
2 df = readExcel(filename) #xlsx ファイルを読み、panda.DataFrame 形式にする  
3 #数学と理科の平均値を求める  
4 #a = mathAndSci(df)  
5 #print(a)  
6 #成績処理  
7 #personalSum(df)  
8 #subjectAverage(df)  
9 #subjectStd(df)  
10 #print(df)
```

Pandas の DataFrame や Series は、Excel の関数でできるようなデータの操作のいくつかを実行することができます。始めに、二つの Series に対して、各項の平均を計算しましょう。通常であれば、二つの Series の一つ一つを対応付けながら和をとって、2 で割る必要があります。しかし、Pandas では、二つの Series を直接的に和ととることができます。ソースコード 4.2 を見てください。二つの列 data['Math'] と data['Science'] に対して、直接和をとって、更に 2 で除しています。結果の a は、Series です。main 部分の 4 行目と 5 行目のコメントを外して、動作を確かめましょう。

ソースコード 4.2 数学と理科の平均値

```
1 def mathAndSci(data:pandas.DataFrame)->pandas.Series:  
2     a = ( data['Math'] + data['Science'] ) / 2  
3     return a
```

もしも、このような機能が使えない場合には、ソースコード 4.3 のように書く必要があります。2 行目で、すべて 0 が入った Series を生成しています。index として、data.index を指定することで、各人に対応させ、一人一人に対して、数学と理科の成績の平均を登録しています。

ソースコード 4.3 数学と理科の平均値

```
1 def mathAndSci(data:pandas.DataFrame)->pandas.Series:  
2     a = pandas.Series([0.] * len(data.index),index = data.index)  
3     for ind in data.index:  
4         a[ind] = ( 1.*data['Math'][ind] + data['Science'][ind]) / 2  
5     return a
```

次に、各人の総点を計算しましょう。DataFrame の sum() というメソッドを使います。axis という引数には、index か columns を指定します。ソースコード 4.4 では、和をとった値 (Series になる) を、sum という名前の列に保存しています。main 部分の 7 行目と最後の行のコメントを外して、動作を確かめましょう。

各科目の平均値と標準偏差も計算しましょう。平均を計算するには、DataFrame の mean() というメソッドを使います。ソースコード 4.5 では、平均をとった値を、average という行として保存しています。さらに、ソースコード 4.6 では、科目の標準偏差を計算しています。std という行に保存する際に、round() というメソッドを使って小数以下 2 桁に丸めています。main 部分の 8 行目と 9 行目のコメントを外して、動作を確かめま

ソースコード 4.4 各人の総点

```
1 def personalSum(data:pandas.DataFrame):  
2     summary = data.sum(axis = 'columns')  
3     data['sum'] = summary
```

しょう。

ソースコード 4.5 科目の平均

```
1 def subjectAverage(data:pandas.DataFrame):  
2     average = data.mean(axis = 'index')  
3     data.loc['average'] = average
```

ソースコード 4.6 科目の標準偏差

```
1 def subjectStd(data:pandas.DataFrame):  
2     std = data.std(axis = 'index')  
3     data.loc['std'] = std.round(1)
```

課題 1 DataFrame には、中央値を求めるメソッド `median()` があります。使い方は、`mean()` 等と同じです。 `subjectMedian()` という関数を完成させ、各科目の中央値を求め、 `median` という名前の行として追加しなさい。

5 次回

Python には、沢山の機能がありますが、簡単に作図ができることも特色です。次回は、作図の基礎を扱います。