7_data_analysis

2017 年 3 月 16 日

1 Introduction to Data Science with Julia

2 目次

3 続データ分析

ここでは DataFrames を使ってデータ分析するときに便利な関数を紹介します。

```
In [1]: import RDatasets, DataFrames
        anscombe = RDatasets.dataset("datasets","anscombe")
```

```
Out[1]: 11 × 8 DataFrames.DataFrame
        | Row | X1 | X2 | X3 | X4 | Y1    | Y2   | Y3    | Y4   |
        |-----|----|----|----|----|-------|------|-------|------|
        | 1   | 10 | 10 | 10 | 8  | 8.04  | 9.14 | 7.46  | 6.58 |
        | 2   | 8  | 8  | 8  | 8  | 6.95  | 8.14 | 6.77  | 5.76 |
        | 3   | 13 | 13 | 13 | 8  | 7.58  | 8.74 | 12.74 | 7.71 |
        | 4   | 9  | 9  | 9  | 8  | 8.81  | 8.77 | 7.11  | 8.84 |
        | 5   | 11 | 11 | 11 | 8  | 8.33  | 9.26 | 7.81  | 8.47 |
        | 6   | 14 | 14 | 14 | 8  | 9.96  | 8.1  | 8.84  | 7.04 |
        | 7   | 6  | 6  | 6  | 8  | 7.24  | 6.13 | 6.08  | 5.25 |
        | 8   | 4  | 4  | 4  | 19 | 4.26  | 3.1  | 5.39  | 12.5 |
        | 9   | 12 | 12 | 12 | 8  | 10.84 | 9.13 | 8.15  | 5.56 |
        | 10  | 7  | 7  | 7  | 8  | 4.82  | 7.26 | 6.42  | 7.91 |
        | 11  | 5  | 5  | 5  | 8  | 5.68  | 4.74 | 5.73  | 6.89 |
```

各列の平均などは describe を使うと便利でした。

```
In [2]: DataFrames.describe(anscombe)
```

```
X1
Min      4.0
```

```
1st Qu.  6.5
Median   9.0
Mean     9.0
3rd Qu.  11.5
Max      14.0
NAs      0
NA%      0.0%

X2
Min      4.0
1st Qu.  6.5
Median   9.0
Mean     9.0
3rd Qu.  11.5
Max      14.0
NAs      0
NA%      0.0%

X3
Min      4.0
1st Qu.  6.5
Median   9.0
Mean     9.0
3rd Qu.  11.5
Max      14.0
NAs      0
NA%      0.0%

X4
Min      8.0
1st Qu.  8.0
Median   8.0
Mean     9.0
3rd Qu.  8.0
Max      19.0
NAs      0
NA%      0.0%

Y1
Min      4.26
1st Qu.  6.3149999999999995
Median   7.58
Mean     7.500909090909093
```

```
3rd Qu.  8.57
Max      10.84
NAs      0
NA%      0.0%

Y2
Min      3.1
1st Qu.  6.695
Median   8.14
Mean     7.500909090909091
3rd Qu.  8.95
Max      9.26
NAs      0
NA%      0.0%

Y3
Min      5.39
1st Qu.  6.25
Median   7.11
Mean     7.500000000000001
3rd Qu.  7.98
Max      12.74
NAs      0
NA%      0.0%

Y4
Min      5.25
1st Qu.  6.17
Median   7.04
Mean     7.50090909090909
3rd Qu.  8.190000000000001
Max      12.5
NAs      0
NA%      0.0%
```

各列ごとに関数を作用させる場合は colwise を使います
基本文法

```
DataFrames.colwise(function, DataFrame)
```

```
In [3]: DataFrames.colwise(mean, anscombe) # 各列の平均
```

```
Out[3]: 8-element Array{Any,1}:
```

```
          [9.0]
          [9.0]
          [9.0]
          [9.0]
          [7.50091]
          [7.50091]
          [7.5]
          [7.50091]
```

特定の列に対してだけ関数を作用させたい場合は、anscombe[[:X1, :Y1]] のように作用させる列を明示的に指定します。

```
In [4]: DataFrames.colwise(mean, anscombe[[:X1, :Y1]])

Out[4]: 2-element Array{Any,1}:
          [9.0]
          [7.50091]

In [ ]:
```

eachrow, eachcol は、for 文を使って各行・列を抜き出すときに便利です。

同様のことは行番号や列番号を指定しても出来ますが、よくあるバグとして df[i,:] と書くべきところを df[:,i] と書いてしまうといったこともあるので eachrow, eachcol を使ったほうが良いと思います。

```
In [5]: for row in DataFrames.eachrow(anscombe)
            println(row)
        end

DataFrameRow (row 1)
X1  10
X2  10
X3  10
X4  8
Y1  8.04
Y2  9.14
Y3  7.46
Y4  6.58


DataFrameRow (row 2)
X1  8
X2  8
X3  8
X4  8
Y1  6.95
Y2  8.14
Y3  6.77
Y4  5.76
```

```
DataFrameRow (row 3)
X1  13
X2  13
X3  13
X4  8
Y1  7.58
Y2  8.74
Y3  12.74
Y4  7.71

DataFrameRow (row 4)
X1  9
X2  9
X3  9
X4  8
Y1  8.81
Y2  8.77
Y3  7.11
Y4  8.84

DataFrameRow (row 5)
X1  11
X2  11
X3  11
X4  8
Y1  8.33
Y2  9.26
Y3  7.81
Y4  8.47

DataFrameRow (row 6)
X1  14
X2  14
X3  14
X4  8
Y1  9.96
Y2  8.1
Y3  8.84
Y4  7.04

DataFrameRow (row 7)
X1  6
```

```
X2  6
X3  6
X4  8
Y1  7.24
Y2  6.13
Y3  6.08
Y4  5.25

DataFrameRow (row 8)
X1  4
X2  4
X3  4
X4  19
Y1  4.26
Y2  3.1
Y3  5.39
Y4  12.5

DataFrameRow (row 9)
X1  12
X2  12
X3  12
X4  8
Y1  10.84
Y2  9.13
Y3  8.15
Y4  5.56

DataFrameRow (row 10)
X1  7
X2  7
X3  7
X4  8
Y1  4.82
Y2  7.26
Y3  6.42
Y4  7.91

DataFrameRow (row 11)
X1  5
X2  5
X3  5
X4  8
```

```
Y1  5.68
Y2  4.74
Y3  5.73
Y4  6.89
```

```
In [6]: for col in DataFrames.eachcol(anscombe)
            println(col)
        end
```

```
(:X1,[10,8,13,9,11,14,6,4,12,7,5])
(:X2,[10,8,13,9,11,14,6,4,12,7,5])
(:X3,[10,8,13,9,11,14,6,4,12,7,5])
(:X4,[8,8,8,8,8,8,8,19,8,8,8])
(:Y1,[8.04,6.95,7.58,8.81,8.33,9.96,7.24,4.26,10.84,4.82,5.68])
(:Y2,[9.14,8.14,8.74,8.77,9.26,8.1,6.13,3.1,9.13,7.26,4.74])
(:Y3,[7.46,6.77,12.74,7.11,7.81,8.84,6.08,5.39,8.15,6.42,5.73])
(:Y4,[6.58,5.76,7.71,8.84,8.47,7.04,5.25,12.5,5.56,7.91,6.89])
```

```
In [ ]:
```

eachrow を使って試験 1, 2 両方の点数が 60 点以上の学生の ID を抜き出すには次のようになります。

```
In [7]: scores = DataFrames.readtable("../data/scores.csv")
        DataFrames.head(scores)
        ID  = Int[]
        for student in DataFrames.eachrow(scores)
            if student[:exam1] >= 60 && student[:exam2] >= 60
                push!(ID, student[:ID])
            end
        end
        @show ID;
```

```
ID = [4,7,9,12,16,23,26,28,30,32,42,43,45,48,56,57,61,68,69,75,78,79,81,82,83,85,86,88,8
```

```
In [ ]:
```

目次に戻る

4 Kaggle

以上まででデータ分析に必要な基礎は身についたでしょう。これからはより実践的なデータに触れていきましょう。

とはいえ、統計局が出しているデータなどを用いて分析せよと言われても何をどうしたら良いのかわからないと思うので、Kaggle のチュートリアルをやってみましょう。 Kaggle とはデータサイエンティスト達が集

うコンペティションサイトです。問題によっては懸賞金が掛かっており、中には総額数億円掛かっている問題もあります。

今回はそんな Kaggel の中にチュートリアルとしてある Titanic: Machine Learning from Disaster をやってみましょう。Kaggle を利用するには利用登録する必要が有ります。ですが JuliaBox 同様 Sign Up で Google のアカウントを選べばすぐに終わります。

副題に Machine Learning と入っていますが、機械学習を使わなくても参加できるので心配しないでください。

目次に戻る