Rによるデータの標準化

奥村太一 (2015.11.16)

概要

右のデータは、25人の生徒について得られた学力テスト、学習意欲尺度、学校生活適応度尺度、自尊感情尺度それぞれの得点を示したものである。[1] ここでは、**[学力]**、**[意欲]**、**[適応]**、**[自尊]**の4つの変数についてRを用いて標準得点 (z得点) を算出する (データの標準化を行う) 方法を説明する。

	A	B	C	D	E
1	生徒	学力	意欲	適応	自尊
2	1	38	6	14	3
3	2	75	16	17	14
4	3	44	9	14	11
5	4	59	8	13	7
6	5	50	13	17	3

データの読み込み

まず、このデータを data01 として読み込んでおく。[2]

```
data01 <- read.csv(file="/Users/Name/Desktop/regpath.csv",header=T,
                   fileEncoding="Shift-JIS")
data01
```

データの標準化

変数ごとに標準化を行う方法

まず、変数ごとにもとの得点から標準得点への換算を行い、新たなデータセットを作成するという手順でやってみよう。以下がそのプログラム例である。

```
z1 <- (data01$学力-mean(data01$学力))/sd(data01$学力)
z2 <- (data01$意欲-mean(data01$意欲))/sd(data01$意欲)
z3 <- (data01$適応-mean(data01$適応))/sd(data01$適応)
z4 <- (data01$自尊-mean(data01$自尊))/sd(data01$自尊)
```

まず、データの標準化とは、もとの得点からその変数の平均を引いたものを標準偏差SDで割るという手続きであった。つまり、もとの得点をx、平均をM、標準偏差をsと表せば、標準得点zは、以下のようになる。

$$z = \frac{x - M}{s}$$

上記プログラムの各行は、この操作を変数ごとに行い、標準得点を代入する内容となっている。ここで用いられている関数は以下の通りである。

mean()	( )内の変数について、平均を算出する関数
sd()	( )内の変数について、標準偏差を算出する関数

なお、Rでは、加減乗除はそれぞれ +,-,*,/ で表され、項をくくりたい場合は () を利用すればよい。

---

[1] 最初の5名分だけを表示してある。なお、このデータは http://www.juen.ac.jp/lab/okumura/data.html から"regpath.csv" としてダウンロード可能である (リンクを右クリックして保存)。

[2] 外部ファイルの読み込みについては、回帰分析のレジュメを参照のこと。

また、データが格納されたオブジェクト (**データフレーム**という) 内の変数は、「**データフレーム名$変数名**」で指定できる。例えば、data01の[学力]は、data01**$学力** となる。

以上で、z1からz4にそれぞれ[学力]、[意欲]、[適応]、[自尊]の標準得点が格納されたことになる。実際にこれらの平均がゼロ、標準偏差が1となっているか確認してみよう。[3]

```
mean(z1); mean(z2); mean(z3); mean(z4)
sd(z1); sd(z2); sd(z3); sd(z4)
```

これを実行すると、以下のように各変数の平均と標準偏差が縦に並んだ形で表示される。

```
> mean(z1); mean(z2); mean(z3); mean(z4)
[1] 8.158188e-17
[1] 1.981596e-16
[1] -7.830542e-17
[1] -1.032638e-16
> sd(z1); sd(z2); sd(z3); sd(z4)
[1] 1
[1] 1
[1] 1
[1] 1
```

平均のアウトプットがわかりにくいが、eは指数であり ae-b とは $a \times 10^{-b}$ (= $a/10^{b}$) を意味する。つまり、8.158188e-17とは 8.158188/10^{17} ということになり、これは実質ゼロである。

さて、この標準化後の得点と、data01に**[生徒]**として入っていた生徒IDを結合させて、新しいデータフレーム data02 を作ってみよう。以下がそのコマンドである。

```
data02 <- data.frame(生徒=data01$生徒,学力=z1,意欲=z2,適応=z3,自尊=z4)
```

ここでは、データフレームを作成する関数 data.frame が用いられている。関数 data.frame の説明は以下の通り。ここでは、data01の[生徒]はそのまま、z1からz4にはそれぞれ [学力] [意欲] [適応] [自尊] の名をつけてdata02に格納した。

data.frame()	データフレームを作成する関数	
	name1=x1,name2=x2,…	x1,x2,… の各変数に name1,name2,… の名前を付ける

作成した data02 の中身を表示すると、次のようになっている。各行が、それぞれの生徒の４変数の標準得点 (z得点) である。

```
> data02
  生徒       学力       意欲       適応       自尊
1    1 -0.8607157 -1.48315522  0.71053130 -1.42718582
2    2  0.8700713  0.94030102  1.50590217  0.62766287
3    3 -0.5800475 -0.75611835  0.71053130  0.06724959
4    4  0.1216229 -0.99846397  0.44540768 -0.67996811
5    5 -0.2993794  0.21326415  1.50590217 -1.42718582
(…以下省略)
```

[3] セミコロン ; で区切ることで、複数のコマンドを１行に書くことができる。

複数の変数をまとめて標準化する

ここまでは変数ごとにその平均と標準偏差を用いて標準化を行ってきたが、変数の数が多くなると手間が増えて大変である。複数の変数をまとめて標準化する方法はないだろうか。

実は、Rには scale という関数があり、これを用いることで複数の変数をまとめて標準化することができる。

scale()	( )内のデータフレームについて、標準化を行った結果を返す関数。

これを用いて [学力]から[自尊]までの４変数を標準化し、標準得点をzに代入するコマンドは以下の通りである。

```
z <- scale(data01[,2:5])
```

上記の data01[,2:5] は、data01の2〜5列目までを指している。すなわち、今回の例では[学力]から[自尊]までの４変数のデータに相当する。[4] このzの中身を表示させると、以下の通りとなる。

```
> z
          学力        意欲        適応        自尊
 [1,] -0.8607157 -1.48315522  0.71053130 -1.42718582
 [2,]  0.8700713  0.94030102  1.50590217  0.62766287
 [3,] -0.5800475 -0.75611835  0.71053130  0.06724959
 [4,]  0.1216229 -0.99846397  0.44540768 -0.67996811
 [5,] -0.2993794  0.21326415  1.50590217 -1.42718582
(…以下省略)
```

このzについて、各列の平均と標準偏差をまとめて算出してみよう。これには、apply という関数が使える。コマンドと実行結果は以下の通り。

```
> apply(X=z,MARGIN=2,FUN=mean)
        学力         意欲         適応         自尊
 8.158188e-17  1.981596e-16 -7.830542e-17 -1.032638e-16
> apply(X=z,MARGIN=2,FUN=sd)
学力 意欲 適応 自尊
   1    1    1    1
```

関数 apply は、以下のように、特定のオブジェクトについて行方向または列方向の演算を行わせるものである。

apply()	行方向、列方向の演算を行わせる関数。カッコ内の引数の意味は以下の通り。	
	X=	演算を行う対象を指定する。
	MARGIN=	1であれば行(横)方向、2であれば列(縦)方向の演算を行う。
	FUN=	演算を行う関数を指定する。平均ならmean、標準偏差ならsd、など。

関数 scale や apply により一括して計算された標準得点、およびそれらの平均と標準偏差と、変数ごとに算出された結果を比較して、同じであることを確認しよう。

---

[4] 1列目は生徒のIDであるので、標準化の対象にはしない。ちなみに、1列目は data01[,1] でアクセスできる。また、データのうち特定の行 (例えば1〜10行目) にアクセスしたい場合は data01[1:10,] のように指定する。行と列を両方指定する場合 (例えば、最初の10人について変数2〜5までのデータを取り出したい) は、data01[1:10,2:5] のようにすればよい。

さらに、data02を作成したときと同様、標準得点が格納されたzとdata01の生徒ID [生徒] を統合して1つのデータフレーム data03 を作成するには、下記のようにすればよい。

```
data03 <- data.frame(生徒=data01$生徒,z)
```

この data03 の中身を表示すれば、data02 と全く同じであることが確認できるはずである。

標準偏回帰係数の算出

全ての変数を標準化した上で回帰分析を実行した場合、得られた偏回帰係数を**標準偏回帰係数**という。変数間でスケールが異なる場合、これを標準化という形でそろえた上で偏回帰係数の値を比較したいときに用いる。

標準化したデータを用いた回帰分析の実行

以下は、上記で作成した data03 を用いて、[学力]を従属変数、[意欲]と[適応]を独立変数とした重回帰分析を行った結果である。Coefficients: の Estimate に標準偏回帰係数の値が記載されている。この場合、[意欲]と[適応]にかかる標準偏回帰係数は、それぞれ0.470と0.294である。通常の偏回帰係数と同様、検定結果とあわせて

> 「“意欲”の標準偏回帰係数は5%水準で有意であった ($b^* = 0.470$, $t(22)=2.646$, $p = .015$)。」

のように報告する。[5]

```
> fit03 <- lm(学力~意欲+適応, data=data03)
> summary(fit03)
(…一部省略)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -9.252e-17  1.592e-01   0.000   1.0000
意欲         4.700e-01  1.777e-01   2.646   0.0148 *
適応         2.940e-01  1.777e-01   1.655   0.1121
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(…一部省略)
```

なお、変数を全て標準化した場合、切片の値はゼロとなる。[6]

標準偏回帰係数の解釈

標準化後の変数においては、1点は標準偏差1個分ということである。従って、上記のように[意欲]にかかる標準偏回帰係数が0.47であるというのは、

> [適応]が一定のままで、[意欲]のみが1標準偏差 (標準偏差1個分) 高くなった場合、[学力]の予測値は0.47標準偏差 (標準偏差0.47個分) 高くなる。

という意味として解釈すればよいことになる。[7]

---

[5] 標準偏回帰係数は、b^*で表すことが多い。

[6] 切片とは「全ての独立変数の値がゼロの場合の従属変数の予測値」であり、標準化している場合は「全ての独立変数の値がその平均の場合の従属変数の予測値」となる。独立変数が全て平均の場合は、従属変数の予測値もその平均となるのだが、標準化していればこれはゼロであるため、切片はゼロとなる。

[7] ただし、因果関係や個人内での変化を意味しているわけではないことに注意が必要である。