

6日目：データをチェックする

今日は、読み込んだデータのチェックをしてみましょう。また、4日目にやったように、特定のデータだけを抽出して別のデータ（変数）を作ることもやってみたいと思います。これが、あとで結構役に立つのです。

まずはRを起動しますが、今日は昨日作って保存しておいたRエディタのファイルをダブルクリックすることで起動してみてください。

エクセルなどと同様に、RもRで作ったファイルをクリックすることで起動させることができます。そして、このファイルから起動することで、手間をひとつ省くことができます。

Rが起動したら、現在の作業ディレクトリを確認してみてください。すると、昨日のように作業ディレクトリを変更しなくても、Rの練習用のフォルダが作業ディレクトリになっているはず。RエディタのファイルからRを起動した場合、そのファイルのあったところが作業ディレクトリになります。

開いたRエディタには、ファイルを読み込む命令が残っているはずなので、それを使ってデータファイルを読み込んでください。

では、読み込んだデータ（`x` という変数名にしたとします）を簡単にチェックするいくつかのコマンドを紹介します。

- ・すべてを表示させる

x

- ・表の上から3行分だけを表示させる（`head(x)`にすれば自動的に5行分）

head(x,3)

- ・表の下（最後）から3行分だけを表示させる（`tail(x)`にすれば自動的に5行分）

tail(x,3)

- ・変数名（列に付けられた名前）を表示させる

colnames(x)

- ・列数を求める（表示させる）

ncol(x)

- ・行数を求める（表示させる）

nrow(x)

全体的なチェックに使えるのはこのようなものでしょう。

次に、部分的にチェックするのに使えるコマンドを紹介します。引き続き、データのファイル名はxです。

- ・ある変数名のデータを求める（表示させる）（たとえば、「年齢」なら）

x\$年齢

- ・特定のセル入っているデータを求める（表示させる）（たとえば、3行3列目なら）

x[3, 3]

- ・特定の行に入っているデータを求める（表示させる）（たとえば、2行目なら）

x[2,]

- ・特定の列に入っているデータを求める（表示させる）（たとえば、3列目なら）その1

x[, 3]

- ・特定の列に入っているデータを求める（表示させる）（たとえば、3列目なら）その2

x[3]

なお、上のコマンドは、見やすくなるように半角空白を入れている部分もあります。この半角空白は入れなくても問題はありません。

特定のセルの表示、特定の行、特定の列と見比べると、列番号、行番号をはぶくと、その行、もしくは列全体が表示されるわけですが、少し違うのは、x[3]という列表示です。x[, 3]と両方やってみるとわかりますが、x[, 3]は数値を横に表示し、列名は表示されません。x[3]は数値を縦に表示し、変数名が表示されます。

次に、複数の行や列、セルを抽出して、そこだけをチェックすることを考えてみます。4日目にやった、あるベクトルから任意のものを抽出したときのことを思い出してください。その時のやり方を援用すれば、いくつかの方法が考えられます。

まずひとつは、抽出する複数の行や列を「:」でつないで入力することが考えられます。以下のような指示は計算してくれるようです。

x[3, 3:5]

x[2:4,]

x[, 3:5]

x[3:5]

実行した結果を眺めてもらえれば一目瞭然ですが、上から、「3行目の3列目から5列目までのデータ」、「2行目から4行目までのデータ」「3列目から5列目までのデータ」「3列目から5列目までのデータ」が表示されます。また「3列目から5列目までのデータ」であるx[, 3:5]とx[3:5]は、複数列指定の場合は全く同じ出力結果となります。

ところが、このような行列の番号ではなく、変数名を使って複数列をチェックしたい、と

というのは、ちょっと面倒です。ある変数のみであれば、`x$年齢`でよいのですが（これは、ファイル名と変数名を「\$」で結ぶことで指定している形式）、この形式では `x$年齢:b2` といった指定の仕方は受け付けてくれません。以下のように、`c` でくくってズラズラと並べないとダメなようです。なお、文字列をデータとして認識させるためには、「"」でくくってやります。

```
x[c("年齢", "b1", "b2")]
```

しかし、私とかだと、このようなものをいちいち書くのは面倒だ…とってしまうわけです。何とか、もっと簡単にできないか…

そこでこれを、3列目のデータを表示させる `x[3]` という極めて簡単なコマンドと比較してみます。すると、3 のかわりに、`c("年齢", "b1", "b2")` がはいつているということがわかります。逆に言えば、`c()` で変数名のリスト（ベクトル）を作り、たとえば `v` という名前で保存しておけば、`x[v]` でそのリスト内のデータが表示されるということになります（要は、`x[3]` の列番号3の部分、を作成した変数リスト `v` に置きかえたものが `x[v]` というわけです）。

このような変数名を抽出してリストにしておくことは、今後いろいろな時にかなり役立ちますので、覚えておいてください。

ところが、`v <- c("no", "性別"...` などと一から入力するのは、変数が多くなるととても手間… なので、簡単に作る方法を考えます。

先に、`colnames(x)` で変数名が出力されることを紹介しました。その際、変数名は `"no"` というように表示されたはずですが、この出力を使えば、変数名を書き、さらに「"」でくくるといった作業をしなくて済みます。

そこでまず `colnames(x)` で変数名を出力させます。その結果を、コピー&ペーストで R エディタに貼り付けます。こうしておいて、後はそれを加工すればよいわけです（メニューバーの「編集」の中にある、検索、置換をうまく使えば、さらに簡単！）。

```
5 # "no" "性別" "年齢" "b1" "b2" "b3" "b4" "b5"
5 # ↑これをもとに、空白を消し、カンマを入れ、c()でくくると簡単に以下ができます。
7 v <- c("no", "性別", "年齢", "b1", "b2", "b3", "b4", "b5")
8 # ↑これをもとに、必要な変数だけを残したり入れ替えたりすれば、いろいろな変数リストが簡単できます。
9 v2 <- c("b1", "b2", "b3", "b4", "b5")
9 v3 <- c("性別", "b2", "b4", "no")
```

上の図の `v` だけを残しておけば、いつでも簡単に必要な変数だけを取り出した新しいファ

イルを作成することができます。

ちなみに、ここで **x[v3]** と命令してみると…

```
> # "no" "性別" "年齢" "b1" "b2" "b3" "b4" "b5"
> # ↑これをもとに、空白を消し、カンマを入れ、c()でくると簡単に以下ができます。
> v <- c("no", "性別", "年齢", "b1", "b2", "b3", "b4", "b5")
> # ↑これをもとに、必要な変数だけを残したり入れ替えたりすれば、いろいろな変数リストが簡単できます。
> v2 <- c("b1", "b2", "b3", "b4", "b5")
> v3 <- c("性別", "b2", "b4", "no")
> x[v3]
  性別 b2 b4 no
1     2  5  4 1001
2     2  2  3 1002
3     2  4  4 1003
4     2  3  4 1004
5     1  5  5 1005
6     1  3  5 1006
7     2  5  4 1007
8     2  3  3 1008
9     2  4  4 1009
10    1  4  3 1010
11    1  3  4 1011
12    1  4  3 1012
13    2  3  4 1013
14    2  3  4 1014
15    2  5  5 1015
16    2  5  5 1016
17    2  3  3 1017
18    2  4  4 1018
19    1  4  4 1019
20    2  5  5 1020
>
```

当然ですが、このような出力となります。

さて、この **v3** ですが、もちろんその中身は、変数名（だけ）です。それらの列のデータを伴っているものではありません。 **x[v3]** というシンプルな命令を見ていると、 **x[v3]** という名前のデータがあるかのようにも思えてきますが、これは「**x** から **v3** にある変数名の行を取り出せ」という命令なのです。 **v3** で指定された変数名だけからなる新しいデータファイルを作るには、もう一手間かけておく必要があります。一手間といっても、以下のように新しい変数名を用意し、 **x[v3]** の結果を代入するだけです。

```
x3 <- x[v3]
```

こうしておいて **x3** の中身を見ると、次の図のようなデータファイルになっていることが確認できるはずです。

これで6日目は終了です。Rエディタの中身は保存しておいてください。

（今日使ったコマンドでたくさん遊んでみるといいと思います。特に行や列に馴染みのない人は、これらのコマンドに慣れるのに少し時間がかかると思うので…）

```
>
> v3
[1] "性別" "b2"  "b4"  "no"
> x3 <- x[v3]
> x3
  性別 b2 b4  no
1     2  5  4 1001
2     2  2  3 1002
3     2  4  4 1003
4     2  3  4 1004
5     1  5  5 1005
6     1  3  5 1006
7     2  5  4 1007
8     2  3  3 1008
9     2  4  4 1009
10    1  4  3 1010
11    1  3  4 1011
12    1  4  3 1012
13    2  3  4 1013
14    2  3  4 1014
15    2  5  5 1015
16    2  5  5 1016
17    2  3  3 1017
18    2  4  4 1018
19    1  4  4 1019
20    2  5  5 1020
> |
```