第4回 データフレームの基本操作 その2（解答付き）

2023年3月17日

目 次

1 ここで学ぶ事

- データフレームの要素へのアクセス方法
- データの条件抽出
- データフレームのデータの書き換え方法

2 データフレームの要素へのアクセス

プロンプトでデータフレームを代入した変数を評価すると，データフレーム全体が表示された．しかし何千ものデータを格納したデータフレーム全体を表示させるのは効率的ではない．大きなデータセットの中身を確認する場合には，始めの数行を表示させたり，特定の列だけを表示させたりしたいであろう．また，各データの値を用いて計算する場合には，データフレーム内の要素に直接アクセスする必要がある．ここでは，データフレーム内のデータにアクセスする様々な方法を学ぶ．

2.1 要素番号（インデックス）によるアクセス

- ベクトルの 3 番目の要素にアクセスするとき [3] と書いた．データフレームの 3 行 2 列目にアクセスするには [3, 2] と書く．

- ブラケット [] による要素の指定方法は [行, 列]

- どちらかのインデックスだけ指定することにより，行，または，列だけを抽出することもできる．

```
> personal                                          # 前回作成した personal を使う
        名前  身長 体重 年齢 性別   出身地
ID1 山田太郎 173.5   66   19   男    福岡県
ID2 西南花子 166.4   58   20   女 鹿児島県
ID3 市東治子 168.0   NA   18   女    千葉県
ID4 三宅信二 170.3   81   20   男    岡山県
ID5 山村英二 169.0   61   23   男    福岡県
ID6 平野康介 159.0   63   20   男 神奈川県
> personal[3, 2]                                    # 3 行 2 列目の要素にアクセス
[1] 168
> personal[3, ]                                     # 3 行目だけを抽出
        名前 身長 体重 年齢 性別 出身地
ID3 市東治子  168   NA   18   女 千葉県
> personal[ ,1]                                     # 1 列目だけを抽出
[1] "山田太郎" "西南花子" "市東治子" "三宅信二" "山村英二" "平野康介"
```

インデックスにベクトルを渡すことによって複数の行や列を抽出できる．

```
> personal[c(1, 4), ]                               # 1 行目と 4 行目を抽出
        名前  身長 体重 年齢 性別 出身地
ID1 山田太郎 173.5   66   19   男 福岡県
ID4 三宅信二 170.3   81   20   男 岡山県
> personal[1:3, c(1,6)]                             # 1〜3 行目と，1 列目と 6 列目を抽出.
        名前   出身地
ID1 山田太郎   福岡県
```

```
ID2 西南花子 鹿児島県
ID3 市東治子   千葉県
```

2.2 行名・列名によるアクセス

インデックス番号の代わりに，行名，列名を用いても同様にアクセスできる．上の例と同じデータに行名，列名を用いてアクセスしてみる．

```
> personal["ID3", "身長"]                                    # 「ID3」行の「身長」列を抽出
[1] 168
> personal["ID3", ]                                          # 「ID3」行を抽出
        名前 身長 体重 年齢 性別 出身地
ID3 市東治子  168   NA   18   女 千葉県
> personal[ ,"名前"]                                         # 「名前」列を抽出
[1] "山田太郎" "西南花子" "市東治子" "三宅信二" "山村英二" "平野康介"
> personal[c("ID1", "ID4"), ]                                # 「ID1」と「ID4」行を抽出
        名前  身長 体重 年齢 性別 出身地
ID1 山田太郎 173.5   66   19   男 福岡県
ID4 三宅信二 170.3   81   20   男 岡山県
> personal[paste("ID", 1:3, sep=""), c("名前", "出身地") ]
        名前   出身地
ID1 山田太郎   福岡県
ID2 西南花子 鹿児島県
ID3 市東治子   千葉県
```

一つの列だけにアクセスする場合は,「$」の後に列名を記述してアクセスできる．ただし列名をダブルクォーテションで囲んではいけない．

```
> personal$名前                                              # 列名はダブルクォーテションで囲まない！
[1] "山田太郎" "西南花子" "市東治子" "三宅信二" "山村英二" "平野康介"
> personal$性別
[1] "男" "女" "女" "男" "男" "男"
```

第 3 回の講義ノートでも触れたが，データフレームの実体はリストというデータ構造である.「$」を使ったアクセス方法は，リストの要素（スロット）にアクセスするシンタックス（表記法）である．

2.3 真偽値によるアクセス

インデックス番号や行名，列名の代わりに，真偽値（TRUE, FALSE）を用いて抽出することもできる．真偽値のベクトルをブラケット [] に渡すと，TRUE の位置に対応する要素を抽出する．TRUE は T と書いても，FALSE は F と書いても同じ．

```
> personal[c(T, F, T, F, F, F),]                             # TRUE の位置の行だけが抽出される
        名前  身長 体重 年齢 性別 出身地
```

```
ID1 山田太郎 173.5   66   19   男 福岡県
ID3 市東治子 168.0   NA   18   女 千葉県
> personal[, c(T, F, T, F, T, F)]                          # 1, 3, 5 列目を TRUE で抽出
        名前 体重 性別
ID1 山田太郎   66   男
ID2 西南花子   58   女
ID3 市東治子   NA   女
ID4 三宅信二   81   男
ID5 山村英二   61   男
ID6 平野康介   63   男
> personal[c(T, F, T, F, F, F), c(T, F, T, F, T, F)]    # 上の 2 つの条件を合わせる
        名前 体重 性別
ID1 山田太郎   66   男
ID3 市東治子   NA   女
```

このように真偽値を用いて要素にアクセスする方法はデータフレームだけでなく**ベクトル**でも使える.

```
> a <- c("a", "b", "c", "d")
> a[c(TRUE, FALSE, TRUE, FALSE)]
[1] "a" "c"
```

2.4 チェックリスト

- データフレームの要素をインデックス番号で指定する方法.
- データフレームの要素を行名と列名で指定する方法.
- データフレームの行を抽出する 2 つの方法.
- データフレームの列を抽出する 3 つの方法.
- 真偽値を用いてデータフレームの要素を抽出する方法.

3 条件式

真偽値を用いて要素にアクセスする方法は，条件式を用いて要素を抽出したい場合に威力を発揮する．まず条件式の使い方を学び，その後，ベクトルやデータフレームから条件にマッチするデータを抽出する方法を学ぶ.

3.1 基本的な条件式

まず基本的な条件判断を学ぼう.

```
> 3 > 1                          # 大小関係の比較
[1] TRUE
> 3 < 1
[1] FALSE
> 3 < 3
[1] FALSE
> 3 <= 5
[1] TRUE
> 5 <= 3
[1] FALSE
> 3 <= 3
[1] TRUE
> 3 == 3                         # 等しいかチェック
[1] TRUE
> "3 月" == "4 月"               # 文字列が等しいかも「==」でテスト可能
[1] FALSE
> "3 月" == "3 月"
[1] TRUE
> 3 != 3                         # 等しくないかをテスト
[1] FALSE
> 3 != 1                         # 等しくなければ TRUE，等しければ FALSE
[1] TRUE
> "3 月" != "4 月"
[1] TRUE
> "3 月" != "3 月"
[1] FALSE
> NA == NA                       # 欠損値のチェック
[1] NA
> is.na(NA)                      # is.na() 関数も使える
[1] TRUE
> NA == 4
[1] NA
> is.na(4)
[1] FALSE
> (10 %% 2) == 0                 # 余りが 0 と等しいかで偶数をテスト
[1] TRUE
> (9  %% 2) == 0
[1] FALSE
```

R の基本データ構造はベクトルで，四則演算等の演算子は全てベクトルの要素に対して適用できることを第 1 回に学んだ．条件式も要素ごとに適用される．

```
> a <- 1:10
> b <- 2:11
> a < b                                  # a と b の同じ位置の要素同士の大小比較を行う
 [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
> a == b                                 # a と b の同じ位置の要素同士が等しいかどうかをテスト
 [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
> (a %% 2) == 0                          # a の各要素が偶数かどうかをテスト
```

```
 [1] FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE
```

3.2 条件式を用いた要素の抽出

上で学んだ条件式を用いて，まずはベクトルの要素を抽出してみよう．最後にデータフレームの要素を条件式を用いて抽出する方法を示す．

先ほど作成した 1 から 10 が入ったベクトル a から偶数データだけを取り出す方法を示す．

```
> a
 [1]  1  2  3  4  5  6  7  8  9 10
> (a %% 2) == 0                    # a のうち偶数の要素部分だけ TRUE となる真偽値ベクトル
 [1] FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE
> a[(a %% 2) == 0]                 # 上の条件式を [] に書けば偶数値の要素を抽出できる．
[1]  2  4  6  8 10
```

今度は −10 から 10 まで入ったベクトル b を使って，負の要素を抽出してみよう．

```
> b <- -10:10                                  # b に-10 から 10 までの数字のベクトルを代入
> b
 [1] -10  -9  -8  -7  -6  -5  -4  -3  -2  -1   0   1   2   3   4   5   6   7   8   9
[21]  10
> b < 0                                        # b が負の要素部分が TRUE となる真偽値ベクトル
 [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE
[14] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
> b[b<0]                                       # 真偽値ベクトルを [] に渡して負の要素を抽出
 [1] -10  -9  -8  -7  -6  -5  -4  -3  -2  -1
```

上の例では条件式を直接 [] に書いたが，条件式の結果を変数に代入しておいて，変数を [] に渡しても同じ．

```
> negative <- b < 0                            # b<0 の結果を negative に代入．
> b[negative]                                  # 負の要素を抽出
 [1] -10  -9  -8  -7  -6  -5  -4  -3  -2  -1
```

次はデータフレームから条件にマッチしたデータを抽出してみる．まず女性のデータだけ抽出してみよう．

```
> personal[, "性別"] == "女"          #「性別」列に条件式を適用すると真偽値ベクトルが得られる
[1] FALSE  TRUE  TRUE FALSE FALSE FALSE
> personal[personal[, "性別"] == "女", ]          # 上の条件式を [] に書いて女性の行を抽出
        名前   身長 体重 年齢 性別    出身地
ID2 西南花子 166.4   58   20   女 鹿児島県
ID3 市東治子 168.0   NA   18   女   千葉県
```

上と同じことを，条件式の結果を変数に格納して用いてみる．

```
> is.women <- personal$性別 == "女"                # $記号で列にアクセスしても同じ
> personal[is.women, ]                            # 条件式の結果が入った変数を渡す
        名前   身長 体重 年齢 性別    出身地
ID2 西南花子 166.4   58   20   女 鹿児島県
ID3 市東治子 168.0   NA   18   女    千葉県
```

「西南花子」の出身地を抽出してみる.

```
> personal[personal$名前=="西南花子", "出身地"]  # 条件式で行を特定化し，列名で出身地を指定
[1] "鹿児島県"
```

3.3 実習

体重に欠損値が含まれるデータ（行）を抽出せよ.

```
> personal[is.na(personal$体重), ]
        名前 身長 体重 年齢 性別 出身地
ID3 市東治子  168   NA   18   女 千葉県
```

3.4 複合条件式

複数の条件式を同時に使いたい場合は，論理演算によって条件式をつなげる.

3.4.1 論理和

条件 1 **__または（OR）__** 条件 2

OR 演算子|の例.

```
> TRUE | FALSE
[1] TRUE
> FALSE | TRUE
[1] TRUE
> TRUE | TRUE
[1] TRUE
> FALSE | FALSE
[1] FALSE
```

出身地が福岡県かまたは鹿児島県のデータ（行）を抽出する．条件を括弧で括ることによって，「|」より先に条件式が評価することを明示している[1]．

```
> personal[(personal$出身地 == "福岡県") | (personal$出身地 == "鹿児島県"),]
        名前  身長 体重 年齢 性別    出身地
ID1 山田太郎 173.5   66   19   男    福岡県
ID2 西南花子 166.4   58   20   女 鹿児島県
ID5 山村英二 169.0   61   23   男    福岡県
```

3.4.2 論理積

条件 1 **かつ（AND）** 条件 2

AND 演算子&の例．

```
> TRUE & FALSE
[1] FALSE
> FALSE & TRUE
[1] FALSE
> TRUE & TRUE
[1] TRUE
> FALSE & FALSE
[1] FALSE
```

身長が 170cm 以上の男性を抽出する．

```
> personal[(personal$性別 == "男") & (personal$身長 >= 170), ]
        名前  身長 体重 年齢 性別 出身地
ID1 山田太郎 173.5   66   19   男 福岡県
ID4 三宅信二 170.3   81   20   男 岡山県
```

3.4.3 否定

NOT 演算子!の例．

```
> !TRUE
[1] FALSE
> !FALSE
[1] TRUE
> !("abc" == "abc")
[1] FALSE
```

[1]演算子には優先順位があって,「==」の方が「|」より優先順位が高いので，この場合括弧をつけなくても正しく評価される．しかし，優先順位を間違えると予期しない結果になるので，括弧を用いる習慣をつけよう

出身地が福岡県以外の人を抽出する.

```
> personal[!(personal$出身地 == "福岡県"), ]      # 否定演算子を使って抽出
        名前  身長 体重 年齢 性別   出身地
ID2 西南花子 166.4   58   20   女 鹿児島県
ID3 市東治子 168.0   NA   18   女   千葉県
ID4 三宅信二 170.3   81   20   男   岡山県
ID6 平野康介 159.0   63   20   男 神奈川県
> personal[personal$出身地 != "福岡県", ]         # この場合は「!=」を使うこともできる
        名前  身長 体重 年齢 性別   出身地
ID2 西南花子 166.4   58   20   女 鹿児島県
ID3 市東治子 168.0   NA   18   女   千葉県
ID4 三宅信二 170.3   81   20   男   岡山県
ID6 平野康介 159.0   63   20   男 神奈川県
```

3.5 チェックリスト

- 欠損値をテストする 2 つの方法.
- 偶数かどうかをテストする方法.
- 整数が入ったベクトル a から，正の数を抽出する方法.
- meibo データフレームの中から,「TEL」列の値が「092-331-3921」の行を抽出するコードは？
- NOT 演算子は？
- AND 演算子の意味と記号は？
- OR 演算子の意味と記号は？

4 データフレームの要素の書き換え

これまでデータの抽出（アクセス）方法を学んできた．最後にデータの書き換え方法を学んでおこう.

R でのデータの書き換えは一貫していて，書き換えたい部分へのアクセスを記述し，そこに代入演算子「<-」を用いて上書きしたいデータを代入すれば良い．以下例を示す.

personal データフレームの市東治子の体重は NA であった．ここに 60 を設定した場合は，まず市東治子の体重にアクセスし，そこに代入すれば良い.

```
> personal[personal$名前 == "市東治子", "体重"] <- 60        # アクセス先に代入するだけ
> personal[personal$名前 == "市東治子",]                     # 書き換え成功
        名前 身長 体重 年齢 性別 出身地
ID3 市東治子  168   60   18   女 千葉県
```

一括して変更することもできる．男を全て M に，女を全て F に置き換えてみよう.

```
> personal[personal$性別 == "男", "性別"] <- "M"          # 男を M に置き換える
> personal
        名前  身長 体重 年齢 性別   出身地
ID1 山田太郎 173.5   66   19    M   福岡県
ID2 西南花子 166.4   58   20   女 鹿児島県
ID3 市東治子 168.0   60   18   女   千葉県
ID4 三宅信二 170.3   81   20    M   岡山県
ID5 山村英二 169.0   61   23    M   福岡県
ID6 平野康介 159.0   63   20    M 神奈川県
> personal[personal$性別 == "女", "性別"] <- "F"          # 女を F に置き換える
> personal
        名前  身長 体重 年齢 性別   出身地
ID1 山田太郎 173.5   66   19    M   福岡県
ID2 西南花子 166.4   58   20    F 鹿児島県
ID3 市東治子 168.0   60   18    F   千葉県
ID4 三宅信二 170.3   81   20    M   岡山県
ID5 山村英二 169.0   61   23    M   福岡県
ID6 平野康介 159.0   63   20    M 神奈川県
```

この様な値の再設定方法はデータフレームの要素に限らず，あらゆるところで用いられる．例えば，前回の講義で列名を変更するときには，

```
colnames(personal) <- 新しい列名ベクトル
```

で変更したことを思い出そう．

4.1 チェックリスト

- データフレームの要素を書き換える方法.
- mydata データフレームの 3 行 2 列目に 10 を設定するコードは？

5 宿題

現在，personal の内容は以下の通りである．このデータフレームを用いて以下の問いに答えよ．

```
> personal
        名前  身長 体重 年齢 性別   出身地
ID1 山田太郎 173.5   66   19    M   福岡県
ID2 西南花子 166.4   58   20    F 鹿児島県
ID3 市東治子 168.0   60   18    F   千葉県
ID4 三宅信二 170.3   81   20    M   岡山県
ID5 山村英二 169.0   61   23    M   福岡県
ID6 平野康介 159.0   63   20    M 神奈川県
```

(1) スコアの列を加える．

6 人が試験で以下の得点を得た．このデータを personal の「得点」列に設定せよ．

氏名	得点
山田太郎	96
西南花子	49
市東治子	100
三宅伸治	75
山村英二	81
平野康介	55

解答

```
> 得点 <- c(96, 49, 100, 75, 81, 55)
> personal <- cbind(personal, 得点)
> personal
        名前   身長 体重 年齢 性別    出身地 得点
ID1 山田太郎 173.5   66   19    M    福岡県   96
ID2 西南花子 166.4   58   20    F 鹿児島県   49
ID3 市東治子 168.0   60   18    F    千葉県  100
ID4 三宅信二 170.3   81   20    M    岡山県   75
ID5 山村英二 169.0   61   23    M    福岡県   81
ID6 平野康介 159.0   63   20    M 神奈川県   55
```

(2) スコアをもとに合否判定の列を追加する．

60 点以上の人が合格である．合格なら TRUE を落第なら FALSE を「合否判定」列に設定せよ．ただし，合否判定は条件式を用いて行うこと．

解答

```
> personal <- cbind(personal, 合否判定=(personal$得点 >= 60))
> personal
        名前   身長 体重 年齢 性別    出身地 得点 合否判定
ID1 山田太郎 173.5   66   19    M    福岡県   96     TRUE
ID2 西南花子 166.4   58   20    F 鹿児島県   49    FALSE
ID3 市東治子 168.0   60   18    F    千葉県  100     TRUE
ID4 三宅信二 170.3   81   20    M    岡山県   75     TRUE
ID5 山村英二 169.0   61   23    M    福岡県   81     TRUE
ID6 平野康介 159.0   63   20    M 神奈川県   55    FALSE
```

cbind() を使わずに以下の方法でも新たに列を追加できる．

```
> personal[, "合否判定"] <- personal$得点 >= 60
```

上の方法と cbind() の違いは，cbind() は列を追加した**新たなデータフレーム**を生成し元のデータフレームは不変だが，上の方法は personal の中身を書き換えてしまう点にある．

今のところはどちらでも良いが，関数型プログラミングと呼ばれるプログラミング手法でプログラムを書く場合は中身の書き換えは推奨されない．

(3) 合格者を抽出する．
合格者の行を抽出し一覧表を作成せよ．

解答

```
> personal[personal$合否判定,]
        名前   身長 体重 年齢 性別 出身地 得点 合否判定
ID1 山田太郎 173.5   66   19    M 福岡県   96     TRUE
ID3 市東治子 168.0   60   18    F 千葉県  100     TRUE
ID4 三宅信二 170.3   81   20    M 岡山県   75     TRUE
ID5 山村英二 169.0   61   23    M 福岡県   81     TRUE
```

(4) 列の順番を入れ替える．
列を「名前，得点，合否判定，性別，出身地，身長，体重」に並び替えたデータフレーム personal2 を作成せよ．

解答

```
> personal[, c("名前","得点","合否判定","性別","出身地","身長","体重")]
        名前 得点 合否判定 性別   出身地  身長 体重
ID1 山田太郎   96     TRUE    M   福岡県 173.5   66
ID2 西南花子   49    FALSE    F 鹿児島県 166.4   58
ID3 市東治子  100     TRUE    F   千葉県 168.0   60
ID4 三宅信二   75     TRUE    M   岡山県 170.3   81
ID5 山村英二   81     TRUE    M   福岡県 169.0   61
ID6 平野康介   55    FALSE    M 神奈川県 159.0   63
> personal[, c(1, 7, 8, 5, 6, 2, 3)]  # 列番号でも可能だが表の構造が変わると対応できない
        名前 得点 合否判定 性別   出身地  身長 体重
ID1 山田太郎   96     TRUE    M   福岡県 173.5   66
ID2 西南花子   49    FALSE    F 鹿児島県 166.4   58
ID3 市東治子  100     TRUE    F   千葉県 168.0   60
ID4 三宅信二   75     TRUE    M   岡山県 170.3   81
ID5 山村英二   81     TRUE    M   福岡県 169.0   61
ID6 平野康介   55    FALSE    M 神奈川県 159.0   63
```

(5) 成績付けする．
90 点以上は A，80 点以上は B，60 点以上は C，60 点未満は D の成績を保持する「成績」列を作成せよ．ただし，評価付けは条件式を用いて行うこと．

解答

```
> personal[personal$得点>=90, "成績"] <- "A"          # 成績列は新たに作られる
> personal    # 途中経過を表示
        名前  身長 体重 年齢 性別   出身地 得点 合否判定 成績
ID1 山田太郎 173.5   66   19    M   福岡県   96     TRUE    A
ID2 西南花子 166.4   58   20    F 鹿児島県   49    FALSE <NA>
ID3 市東治子 168.0   60   18    F   千葉県  100     TRUE    A
ID4 三宅信二 170.3   81   20    M   岡山県   75     TRUE <NA>
ID5 山村英二 169.0   61   23    M   福岡県   81     TRUE <NA>
ID6 平野康介 159.0   63   20    M 神奈川県   55    FALSE <NA>
> personal[personal$得点<90 & personal$得点>=80, "成績"] <- "B"
> personal[personal$得点<80 & personal$得点>=60, "成績"] <- "C"
> personal[personal$得点<60, "成績"] <- "D"
> personal
        名前  身長 体重 年齢 性別   出身地 得点 合否判定 成績
ID1 山田太郎 173.5   66   19    M   福岡県   96     TRUE    A
ID2 西南花子 166.4   58   20    F 鹿児島県   49    FALSE    D
ID3 市東治子 168.0   60   18    F   千葉県  100     TRUE    A
ID4 三宅信二 170.3   81   20    M   岡山県   75     TRUE    C
ID5 山村英二 169.0   61   23    M   福岡県   81     TRUE    B
ID6 平野康介 159.0   63   20    M 神奈川県   55    FALSE    D
```

今のところはあまり気にする必要はないが，上のコードも personal の中身を書き換えているので関数型プログラミングのスタイルではない．

(6) 基本統計量を求める．
得点の平均点（mean()）と標準偏差 sd() を求めよ．

解答

```
> mean(personal$得点)      # 平均 (mean)
[1] 76
> sd(personal$得点)        # 標準偏差 (standard deviation)
[1] 20.84226
> var(personal$得点)       # 分散     (variance)
[1] 434.4
> summary(personal$得点)   # 統計サマリ
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  49.00   60.00   78.00   76.00   92.25  100.00
```