Day 2 : Introduction to Genomic Data Visualization in R

Ambu Vijayan

Bioinformatician

BioLit, Thiruvananthapuram

3 Data frames

3.1 Structure of a data frame

Identify structure of our dataser using class() and dim() functions.

```
class(birthweight)
```

[1] "data.frame"

```
dim(birthweight)
```

[1] 42 18

The “[1]” is not part of the output. It is an index added by R to help you keep track of the values when an operation outputs a large number of values.

Let’s take a look at the contents.

```
birthweight
```

Outputs in the console

Console | Terminal | Background Jobs

R 4.2.1 · D:/Users/ambuv/Downloads/github/docs/

```
> birthweight
     ID birth.date    location length birthweight head.circumference weeks.gestation smoker
1  1107  1/25/1967     General     52        3.23                 36               38     no
2   697   2/6/1967 Silver Hill     48        3.03                 35               39     no
3  1683  2/14/1967 Silver Hill     53        3.35                 33               41     no
4    27   3/9/1967 Silver Hill     53        3.55                 37               41    yes
5  1522  3/13/1967    Memorial     50        2.74                 33               39    yes
6   569  3/23/1967    Memorial     50        2.51                 35               39    yes
7   365  4/23/1967    Memorial     52        3.53                 37               40    yes
8   808   5/5/1967 Silver Hill     48        2.92                 33               34     no
9  1369   6/4/1967 Silver Hill     49        3.18                 34               38    yes
10 1023   6/7/1967    Memorial     52        3.00                 35               38    yes
11  822  6/14/1967    Memorial     50        3.42                 35               38     no
12 1272  6/20/1967    Memorial     53        2.75                 32               40    yes
13 1262  6/25/1967 Silver Hill     53        3.19                 34               41    yes
14  575  7/12/1967    Memorial     50        2.78                 30               37    yes
15 1016  7/13/1967 Silver Hill     53        4.32                 36               40     no
16  792   9/7/1967    Memorial     53        3.64                 38               40    yes
17  820  10/7/1967     General     52        3.77                 34               40     no
18  752 10/19/1967     General     49        3.32                 36               40    yes
19  619  11/1/1967    Memorial     52        3.41                 33               39    yes
20 1764  12/7/1967 Silver Hill     58        4.57                 39               41    yes
```

To view in Source pane of Rstudio

```
View(birthweight)
```

birthweight

Filter

	ID	birth.date	location	length	birthweight	head.circumference	weeks.gestation	smoker	maternal.age	mat
1	1107	1/25/1967	General	52	3.23	36	38	no	31	
2	697	2/6/1967	Silver Hill	48	3.03	35	39	no	27	
3	1683	2/14/1967	Silver Hill	53	3.35	33	41	no	27	
4	27	3/9/1967	Silver Hill	53	3.55	37	41	yes	37	
5	1522	3/13/1967	Memorial	50	2.74	33	39	yes	21	
6	569	3/23/1967	Memorial	50	2.51	35	39	yes	22	
7	365	4/23/1967	Memorial	52	3.53	37	40	yes	26	
8	808	5/5/1967	Silver Hill	48	2.92	33	34	no	26	
9	1369	6/4/1967	Silver Hill	49	3.18	34	38	yes	31	
10	1023	6/7/1967	Memorial	52	3.00	35	38	yes	30	

Showing 1 to 10 of 42 entries, 18 total columns

View coloumn names

```
colnames(birthweight)
```

```
Console   Terminal ×   Background Jobs ×
R 4.2.1 · D:/Users/ambuv/Downloads/github/docs/
> colnames(birthweight)
 [1] "ID"                    "birth.date"            "location"
 [4] "length"                "birthweight"           "head.circumference"
 [7] "weeks.gestation"       "smoker"                "maternal.age"
[10] "maternal.cigarettes"   "maternal.height"       "maternal.prepregnant.weight"
[13] "paternal.age"          "paternal.education"    "paternal.cigarettes"
[16] "paternal.height"       "low.birthweight"       "geriatric.pregnancy"
>
```

Now we know all the data we have.

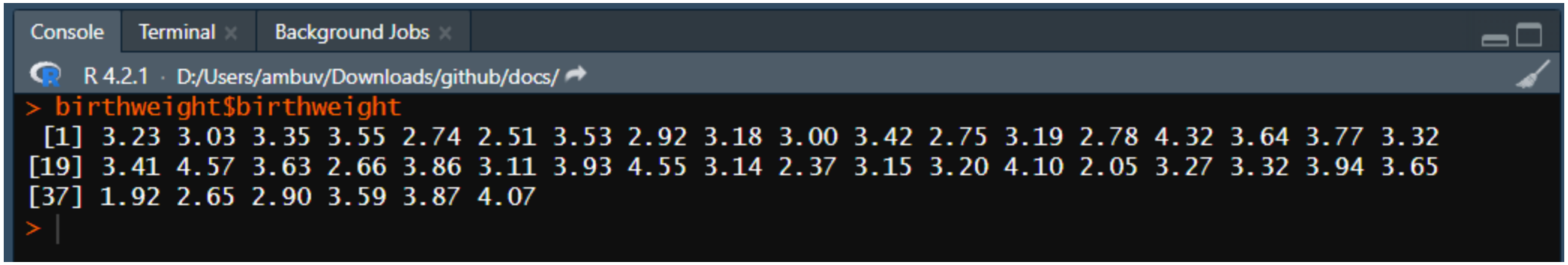
Generally, we don’t want to operate on the entire data frame. For example, to calculate the mean birth weight, we don’t need the information in the “paternal.education” column.

3.2 Selecting a single column using the $ and [[operators

There are three ways to have R subset the data frame: $, [[, and [.

The simplest way to get all the values in the “birthweight” column is with the $ operator.

```
birthweight$birthweight
```



The numbers at the beginning of each line of output give us a general idea of the length of the vector, and allow us to determine the value of a particular observation at a glance.

"what was the birth weight of the 34th baby?"

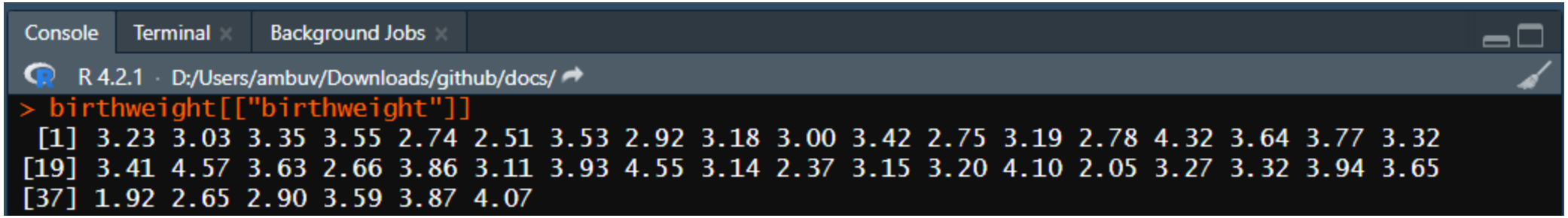
Once the vector of birth weights has been extracted from the rest of the data frame, it can be used to calculate a **mean**.

```
mean(birthweight$birthweight)
```

[1] 3.312857

The $ operator is a shortcut for the [[* *]]. They function in the same way, returning the value of the element named.

```
birthweight[["birthweight"]]
```




```
mean(birthweight[["birthweight"]])
```

[1] 3.312857

Back to coloumnnames

```
colnames(birthweight)
```

```
Console   Terminal   Background Jobs
R 4.2.1 · D:/Users/ambuv/Downloads/github/docs/
> colnames(birthweight)
 [1] "ID"                    "birth.date"            "location"
 [4] "length"                "birthweight"           "head.circumference"
 [7] "weeks.gestation"       "smoker"                "maternal.age"
[10] "maternal.cigarettes"   "maternal.height"       "maternal.prepregnant.weight"
[13] "paternal.age"          "paternal.education"    "paternal.cigarettes"
[16] "paternal.height"       "low.birthweight"       "geriatric.pregnancy"
>
```

```
birthweight[[5]]
```

```
Console   Terminal   Background Jobs
R 4.2.1 · D:/Users/ambuv/Downloads/github/docs/
> birthweight[[5]]
 [1] 3.23 3.03 3.35 3.55 2.74 2.51 3.53 2.92 3.18 3.00 3.42 2.75 3.19 2.78 4.32 3.64 3.77 3.32
[19] 3.41 4.57 3.63 2.66 3.86 3.11 3.93 4.55 3.14 2.37 3.15 3.20 4.10 2.05 3.27 3.32 3.94 3.65
[37] 1.92 2.65 2.90 3.59 3.87 4.07
```

Question

“what was the birth weight of the 25th baby?”

Hint : Use $ and []

Answer

```
birthweight$birthweight[25]
```

[1] 3.93

Mean in a different way

```
mean(birthweight[[5]])
```

[1] 3.312857

Avoid mistakes

 # the $ operator can't take an index

`birthweight$5` will not work

3.3 Selecting a subset of the data frame using the [operator

[returns an object of the same type it is used to subset.
Let's use [to retrieve the fifth column, This will return a data frame with 42 rows and 1 column.

```
birthweight[5]
```

```
birthweight[5]
```

Console Terminal Background Jobs

R 4.2.1 · D:/Users/ambuv/Downloads/github/docs/

```
> birthweight[5]
   birthweight
1         3.23
2         3.03
3         3.35
4         3.55
5         2.74
6         2.51
7         3.53
8         2.92
9         3.18
10        3.00
11        3.42
12        2.75
13        3.19
14        2.78
15        4.32
16        3.64
17        3.77
18        3.32
19        3.41
20        4.57
```

Because the [operator returns a new data frame, it can be used to specify multiple **rows and / or columns**.

```
birthweight[c(1,5)]
```

Console Terminal Background Jobs

R 4.2.1 · D:/Users/ambuv/Downloads/github/docs/

```
> birthweight[c(1,5)]
     ID birthweight
1  1107        3.23
2   697        3.03
3  1683        3.35
4    27        3.55
5  1522        2.74
6   569        2.51
7   365        3.53
8   808        2.92
9  1369        3.18
10 1023        3.00
11  822        3.42
12 1272        2.75
13 1262        3.19
14  575        2.78
15 1016        4.32
16  792        3.64
17  820        3.77
18  752        3.32
19  619        3.41
20 1764        4.57
```

Now what happens if c() is not used?

```
birthweight[1, 5]
```

[1] 3.23

This gives an output of 1st **row** and 5th **column** respectively.

You are basically reading a Matrix!

Lets move into selecting multiple columns and rows

```
birthweight[c(2,7,29), c(1,5)]
```

```
> birthweight[c(2,7,29), c(1,5)]
     ID birthweight
2   697        3.03
7   365        3.53
29 1058        3.15
```

Using a ""-"" before an index or group of indices will exclude the specified rows / columns and "":"" symbol will specify a range.

```
birthweight[c(2,7,29), -c(1:15)]
```

```
> birthweight[c(2,7,29), -c(1:15)]
   paternal.height low.birthweight geriatric.pregnancy
2              178               0               FALSE
7              181               0               FALSE
29             182               0               FALSE
```

R will also accept row or column names in quotations as a way to subset the data frame.

```
birthweight[c("maternal.cigarettes", "birthweight")]
```

Console Terminal Background Jobs

R 4.2.1 · D:/Users/ambuv/Downloads/github/docs/

```
> birthweight[c("maternal.cigarettes", "birthweight")]
   maternal.cigarettes birthweight
1                    0        3.23
2                    0        3.03
3                    0        3.35
4                   25        3.55
5                   17        2.74
6                    7        2.51
7                   25        3.53
8                    0        2.92
9                   25        3.18
10                  12        3.00
11                   0        3.42
12                  50        2.75
13                  35        3.19
14                   7        2.78
15                   0        4.32
16                   2        3.64
17                   0        3.77
18                  12        3.32
19                  25        3.41
20                  12        4.57
```

Finally, vectors of logical (TRUE/FALSE) values can be used to subset data.

Rows or columns corresponding to "TRUE" elements will be returned, while rows or columns corresponding to "FALSE" elements will be excluded.

```
birthweight[c(1,3,5:13), c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE,
TRUE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, TRUE, TRUE)]
```

Console | Terminal | Background Jobs

R 4.2.1 · D:/Users/ambuv/Downloads/github/docs/

```
> birthweight[c(1,3,5:13), c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRU
E, FALSE, FALSE, FALSE, FALSE, TRUE, TRUE)]
     ID birth.date    location length birthweight head.circumference weeks.gestation smoker
1  1107  1/25/1967     General     52        3.23                 36              38     no
3  1683  2/14/1967 Silver Hill     53        3.35                 33              41     no
5  1522  3/13/1967    Memorial     50        2.74                 33              39    yes
6   569  3/23/1967    Memorial     50        2.51                 35              39    yes
7   365  4/23/1967    Memorial     52        3.53                 37              40    yes
8   808   5/5/1967 Silver Hill     48        2.92                 33              34     no
9  1369   6/4/1967 Silver Hill     49        3.18                 34              38    yes
10 1023   6/7/1967    Memorial     52        3.00                 35              38    yes
11  822  6/14/1967    Memorial     50        3.42                 35              38     no
12 1272  6/20/1967    Memorial     53        2.75                 32              40    yes
13 1262  6/25/1967 Silver Hill     53        3.19                 34              41    yes
```

Filtering using logical expressions

```
birthweight$length
```

```
birthweight$length < 50
```

```
> birthweight$length
 [1] 52 48 53 53 50 50 52 48 49 52 50 53 53 50 53 53 52 49 52 58 54 47 52 48 51 56 51 48 53 53
[31] 58 46 51 51 54 53 48 43 53 53 50 53
> birthweight$length < 50
 [1] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
[16] FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE
[31] FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE
```

Since the result of the `birthweight$length < 50` operation is a vector of *TRUE / FALSE* values, it can be used to subset the data frame.

```
birthweight[birthweight$length < 50, c(1,4:12,17,18)]
```

```
> birthweight[birthweight$length < 50, c(1,4:12,17,18)]
     ID length birthweight head.circumference weeks.gestation smoker maternal.age
2   697     48        3.03                 35              39     no           27
8   808     48        2.92                 33              34     no           26
9  1369     49        3.18                 34              38    yes           31
18  752     49        3.32                 36              40    yes           27
22  516     47        2.66                 33              35    yes           20
24  321     48        3.11                 33              37     no           28
28 1363     48        2.37                 30              37    yes           20
32  300     46        2.05                 32              35    yes           41
37  431     48        1.92                 30              33    yes           20
38 1313     43        2.65                 32              33     no           24
```

Observe that the [] function requires 2 components

data[component1, component2]

data[row, columns]

```
birthweight[5, 10]
```

```
birthweight[c(2,7,29), c(1,5)]
```

```
birthweight[birthweight$length < 50, c(1,4:12,17,18)]
```

3.3.1 Subsetting a vector

A vector, like a column of a data frame, can be subsetted using the [] operator with an index or another vector.

```
birthweight$length[1]
```

[1] 52

```
birthweight$length[c(1,2)]
```

[1] 52 48

4 Basic data types

Now we know logical values can be used to subset a data frame, and all the values in a given column of a data frame must be of the same type or class.

4.1 Understanding class

R has the following basic data classes:

- numeric (includes integer and double)
- character
- logical
- complex
- raw

Generally, in bioinformatics, values belong to one of the first three classes.

Numeric

```
class(birthweight$birthweight)
```

[1] "numeric"

Character

```
class(birthweight$smoker)
```

[1] "character"

Logical

```
class(birthweight$geriatric.pregnancy)
```

[1] "logical"

Heads and tails

head and tail commands are used to display the beginning or ending of a data.

```
head(birthweight)
```

```
> head(birthweight)
    ID birth.date    location length birthweight head.circumference weeks.gestation smoker
1 1107  1/25/1967     General     52        3.23                 36               38     no
2  697   2/6/1967 Silver Hill     48        3.03                 35               39     no
3 1683  2/14/1967 Silver Hill     53        3.35                 33               41     no
4   27   3/9/1967 Silver Hill     53        3.55                 37               41    yes
5 1522  3/13/1967    Memorial     50        2.74                 33               39    yes
6  569  3/23/1967    Memorial     50        2.51                 35               39    yes
```

```
head(birthweight$location)
```

```
> head(birthweight$location)
[1] "General"     "Silver Hill" "Silver Hill" "Silver Hill" "Memorial"    "Memorial"
```

Avoid this error : 1 + "1"

The relational operators in R are:

- ">" greater than
- ">=" greater than or equal to
- "<" less than
- "<=" less than or equal to
- "==" equal to
- "!=" not equal to

Example using > relational operators

```
birthweight[birthweight$head.circumference > 35, c("length", "weeks.gestation",
"maternal.height", "paternal.height")]
```

```
> birthweight[birthweight$head.circumference > 35, c("length", "weeks.gestation", "maternal.heigh
t", "paternal.height")]
   length weeks.gestation maternal.height paternal.height
1      52              38             164              NA
4      53              41             161             175
7      52              40             170             181
15     53              40             171             183
16     53              40             170             185
18     49              40             152             170
20     58              41             173             180
21     54              38             172             172
23     52              39             170             178
25     51              38             165              NA
31     58              41             172             185
33     51              40             168             181
34     51              39             157              NA
35     54              42             175             184
42     53              44             174             189
```

Example using <= relational operators

```
birthweight[birthweight$maternal.age <= 20, c("location", "maternal.age",
"paternal.age")]
```

```
> birthweight[birthweight$maternal.age <= 20, c("location", "maternal.age", "paternal.age")]
      location maternal.age paternal.age
11    Memorial           20           22
14    Memorial           19           20
15 Silver Hill           19           19
16    Memorial           20           24
21 Silver Hill           18           20
22 Silver Hill           20           23
26     General           20           23
28     General           20           20
37 Silver Hill           20           20
39     General           19           NA
42 Silver Hill           20           26
```

Notice that when R is asked to perform a comparison between a number and a missing value, the result is a missing value.

```
birthweight[birthweight$paternal.education == 10, c(1,13:16)]
```

```
> birthweight[birthweight$paternal.education == 10, c(1,13:16)]
       ID paternal.age paternal.education paternal.cigarettes paternal.height
NA     NA           NA                 NA                  NA              NA
NA.1   NA           NA                 NA                  NA              NA
7     365           30                 10                  25             181
24    321           39                 10                   0             171
NA.2   NA           NA                 NA                  NA              NA
26   1360           23                 10                  35             179
28   1363           20                 10                  35             185
NA.3   NA           NA                 NA                  NA              NA
36   1191           21                 10                  25             185
37    431           20                 10                  35             180
NA.4   NA           NA                 NA                  NA              NA
```

Exclude all results that is equal to 40

```
birthweight[birthweight$weeks.gestation != 40, "weeks.gestation"]
```

[1] 38 39 41 41 39 39 34 38 38 38 41 37 39 41 38 35 39 37 38 44 41 37 41 41 35
[26] 39 42 42 33 33 39 45 44

Filtering only General

```
birthweight[birthweight$location == "General",]
```

```
> birthweight[birthweight$location == "General",]
     ID birth.date location length birthweight head.circumference weeks.gestation smoker
1  1107  1/25/1967  General     52        3.23                 36              38     no
17  820  10/7/1967  General     52        3.77                 34              40     no
18  752 10/19/1967  General     49        3.32                 36              40    yes
26 1360  2/16/1968  General     56        4.55                 34              44     no
28 1363   4/2/1968  General     48        2.37                 30              37    yes
33 1088  7/24/1968  General     51        3.27                 36              40     no
36 1191   9/7/1968  General     53        3.65                 33              42     no
39 1600  10/9/1968  General     53        2.90                 34              39     no
40  532 10/25/1968  General     53        3.59                 34              40    yes
41  223 12/11/1968  General     50        3.87                 33              45    yes
```

Checking if all values in a column is an integer.

```
is.numeric(birthweight$ID)
```

[1] TRUE

```
is.numeric(birthweight$smoker)
```

[1] FALSE

4.2 Coercion: converting between classes

The birthweight data frame has three columns that should probably be logical values: “smoker”, “low.birthweight”, and “geriatric.pregnancy”.

Only “geriatric.pregnancy” is stored as a logical value. Storing “smoker” and “low.birthweight” as logical values would be more useful, since it allows us to subset the data frame more easily.

Changing the class of data is known as coercion.

```
as.logical(birthweight$low.birthweight)
```

```
> as.logical(birthweight$low.birthweight)
 [1] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[16] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
[31] FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE
```

The coercion rule in R is as follows:

logical > integer > numeric > complex > character

R can convert logical values to integers, store integers as the more general numeric type, or represent numeric data as a character, but these coercion operations cannot always be reversed without losing information.

lets discuss some scenarios below.

Logical to Integer Conversion

```
logical_value <- TRUE
integer_value <- as.integer(logical_value) # Convert TRUE to 1
print(integer_value)
```

Integer to Numeric Conversion

```
integer_data <- 42
numeric_data <- as.numeric(integer_data) # Convert integer to numeric
print(numeric_data)
```

Numeric to Character Conversion

```
numeric_data <- 3.14159
character_data <- as.character(numeric_data) # Convert numeric to character
print(character_data)
```

Character to Numeric (with potential information loss)

```
character_number <- "123.45"
numeric_from_character <- as.numeric(character_number) # Convert character to numeric
print(numeric_from_character)
```

Reversing Character to Numeric (with potential loss)

```
character_text <- "Hello, world!"
numeric_from_text <- as.numeric(character_text) # Attempt to convert text to numeric
print(numeric_from_text) # Output will be NA (Not Available) as text to numeric is not straightforward.
```

Question : convert “smoker” from character to logical

Simple coercion is not going to convert the “smoker” column from character to logical.

How can you solve this problem?

Answer

```
birthweight$smoker == "yes"
```

```
[1] FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE TRUE FALSE TRUE
[13] TRUE TRUE FALSE TRUE FALSE TRUE TRUE TRUE FALSE TRUE TRUE FALSE
[25] FALSE FALSE TRUE TRUE FALSE FALSE FALSE TRUE FALSE TRUE FALSE FALSE
[37] TRUE FALSE FALSE TRUE TRUE FALSE
```

Replacing it in the parent database

```
birthweight$smoker <- (birthweight$smoker == "yes")
```