Session 6: How can I subset data and define missing values?

A Slower Introduction to R

Yea-Hung Chen, PhD, MS
UCSF Library

Thursday, November 13, 2025

Restricting to non-missing values

Systolic versus hypertension

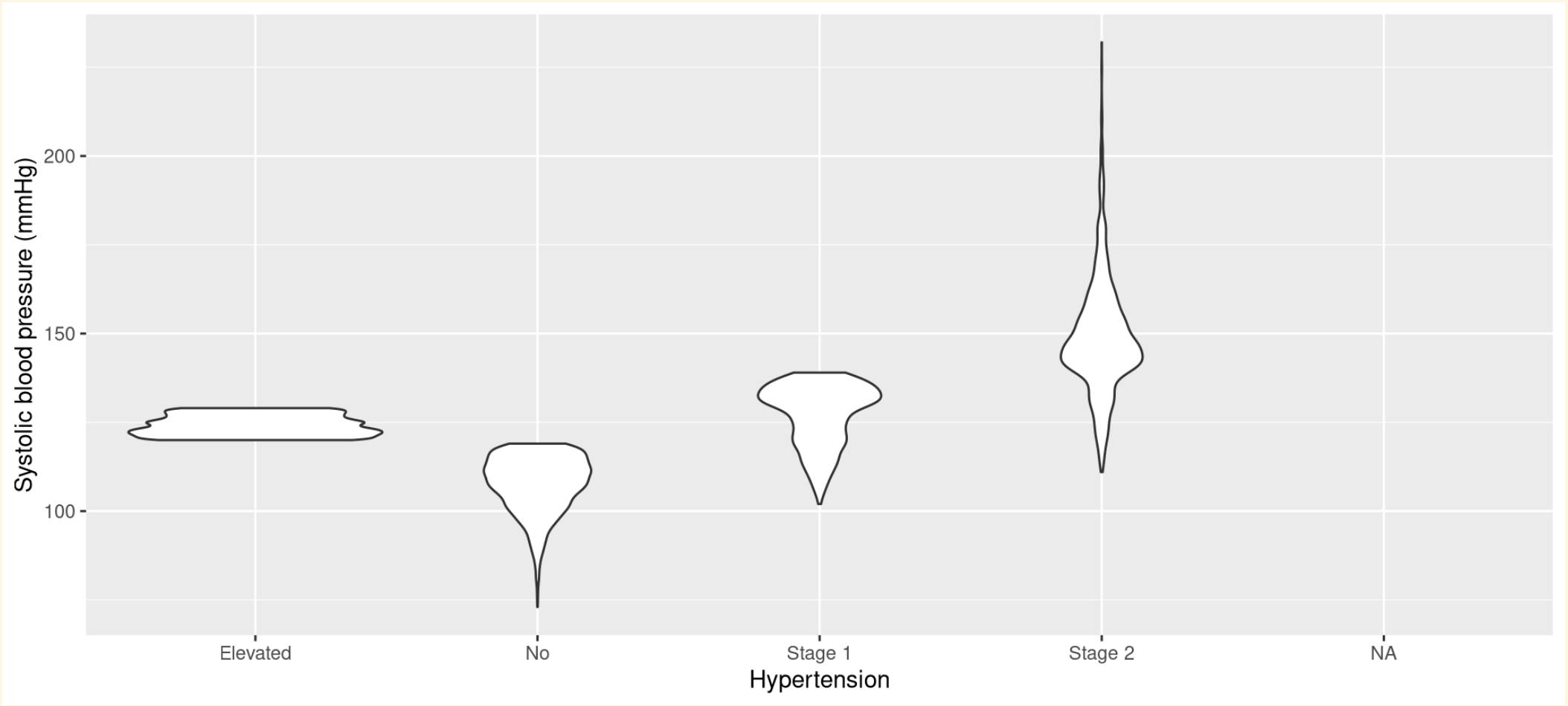
- from Exercise 1, last week:

> Create a violin plot of systolic blood pressure (`bpxosy1`) versus hypertension (`hypertension`), with systolic blood pressure on the *y* axis. You can replace `geom_boxplot()` above with `geom_violin()`. Add labels to the axes using `labs()`. Run your code.

Systolic versus hypertension

```
ggplot(nhanes,aes(x=hypertension,y=bpxosy1))+
  geom_violin()+
  labs(x='Hypertension',y='Systolic blood pressure (mmHg)')
```

Systolic versus hypertension



Terminology

> **Note**
>
> Synonyms of restricting include subsetting and filtering.

Mini data

- as a teaching example, I've created a smaller version of the NHANES data
- as always, I encourage you to focus on watching rather than practicing during the mini lectures
- but, here is code for `minidata` if you want to work through these examples later:

```
minidata<-nhanes[1:10,c('seqn','ridageyr','bpxosy1','bpxodi1','hypertension')]
```

Mini data

```
minidata
#|       seqn ridageyr bpxosy1 bpxodi1 hypertension
#|  1  130378       43     135      98      Stage 2
#|  2  130379       66     121      84      Stage 1
#|  3  130380       44     111      79           No
#|  4  130384       43      NA      NA         <NA>
#|  5  130385       65      NA      NA         <NA>
#|  6  130386       34     110      72           No
#|  7  130387       68     143      76      Stage 2
#|  8  130388       27     130      95      Stage 2
#|  9  130389       59     145      76      Stage 2
#|  10 130390       31     113      78           No
```

Data for individuals not missing hypertension?

- to subset the data to individuals not missing hypertension, use `subset()` and `is.na()`:

```
subset(minidata,!is.na(hypertension))
#|       seqn ridageyr bpxosy1 bpxodi1 hypertension
#|  1  130378       43     135      98      Stage 2
#|  2  130379       66     121      84      Stage 1
#|  3  130380       44     111      79           No
#|  6  130386       34     110      72           No
#|  7  130387       68     143      76      Stage 2
#|  8  130388       27     130      95      Stage 2
#|  9  130389       59     145      76      Stage 2
#|  10 130390       31     113      78           No
```

is.na()

- the values of `hypertension` are:

```
minidata$hypertension
#|   [1] "Stage 2" "Stage 1" "No"      NA        NA        "No"      "Stage 2"
#|   [8] "Stage 2" "Stage 2" "No"
```

- `is.na()` shows whether the values are missing:

```
is.na(minidata$hypertension)
#|   [1] FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
```

- placing an exclamation point in front reverses the values:

```
!is.na(minidata$hypertension)
#|   [1]  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
```

subset()

```
subset(minidata,!is.na(hypertension))
```

- there are two arguments to `subset()` here:
  - `minidata`: the data I want to subset
  - `!is.na(hypertension)`: a rule defining how I want to subset
- `subset()` will keep rows that are `TRUE` in the second argument

subset()

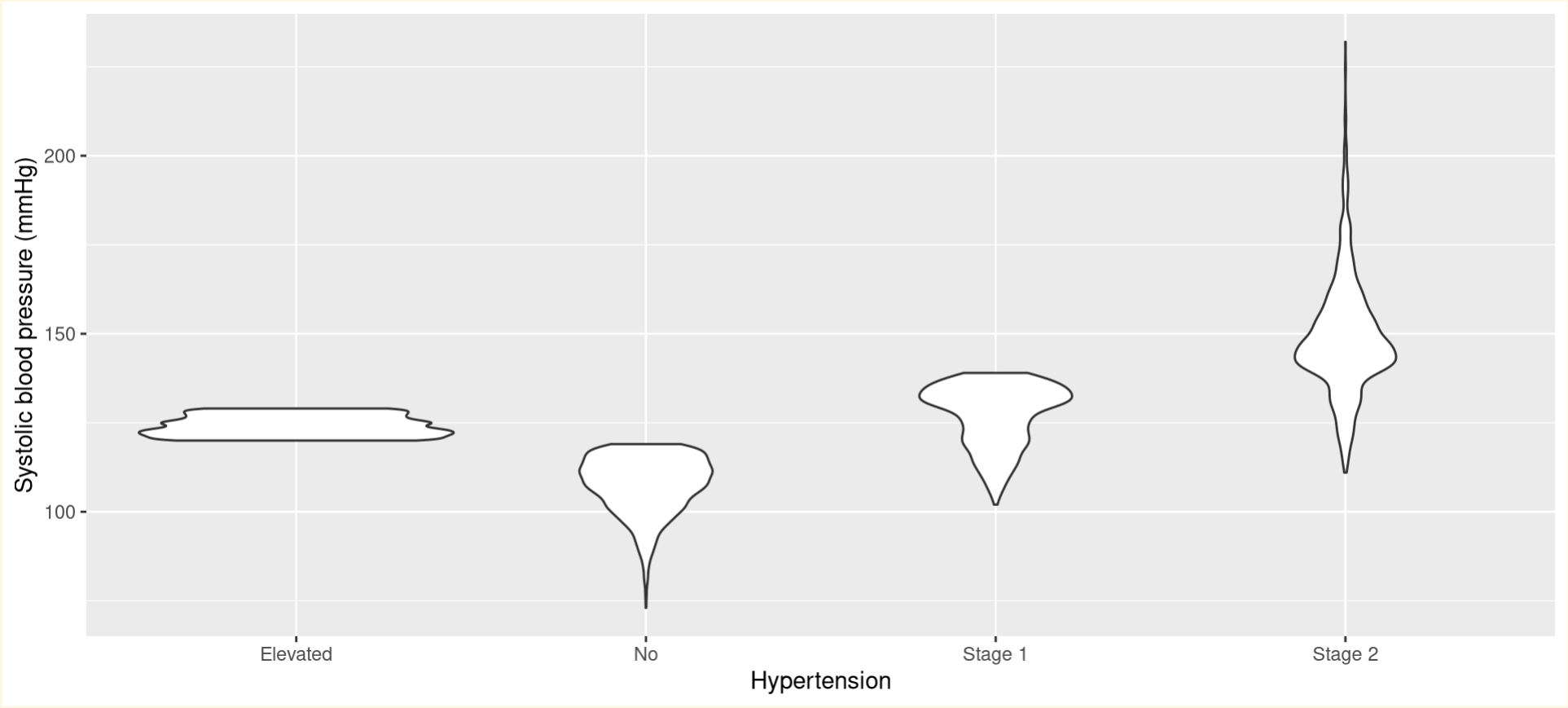
```
subset(minidata,!is.na(hypertension))
```

- dollar-sign notation is not needed in the second argument
- in `subset()`, it is understood that any variable in the second argument is about the data in the first argument

Systolic versus hypertension

```
ggplot(subset(nhanes,!is.na(hypertension)),aes(x=hypertension,y=bpxosy1))+
  geom_violin()+
  labs(x='Hypertension',y='Systolic blood pressure (mmHg)')
```

Systolic versus hypertension



Notice that the groups are still out of order. We'll talk about how to handle this next

Exercise 1

Exercise 1

1. At the top of a new script, include code for loading the `ggplot2` package and importing the data. Run your code.
2. Using `ggplot2`, create density plots of systolic blood pressure (`bpxosy1`), with faceting by `asthma` (defining the rows). In `ggplot()`, define `data` to be a subset of the NHANES data that only includes individuals who are non-missing for `asthma`.

Numerical comparisons

Data for individuals 60 years of age or older?

- suppose I want to look at the data for just individuals who are 60 years of age or older

Data for individuals 60 years of age or older?

- to subset to individuals 60 years of age or older, use `subset()` and a numerical comparison:

```
subset(minidata,ridageyr>=60)
#|      seqn ridageyr bpxosy1 bpxodi1 hypertension
#|   2 130379       66     121      84      Stage 1
#|   5 130385       65      NA      NA         <NA>
#|   7 130387       68     143      76      Stage 2
```

Data for individuals 60 years of age or older?

```
subset(minidata,ridageyr>=60)
```

- there are two arguments to `subset()` here:
  - `minidata`: the data I want to subset
  - `ridageyr>=60`: the rule defining how I want to subset

Numerical comparisons

```
minidata$ridageyr
#|   [1] 43 66 44 43 65 34 68 27 59 31
minidata$ridageyr>=60
#|   [1] FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE
```

Numerical comparisons

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

Numerical comparisons

```
minidata$ridageyr
#|   [1] 43 66 44 43 65 34 68 27 59 31
minidata$ridageyr==43
#|   [1]  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
minidata$ridageyr<59
#|   [1]  TRUE FALSE  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE  TRUE
```

`subset()` and numerical comparisons

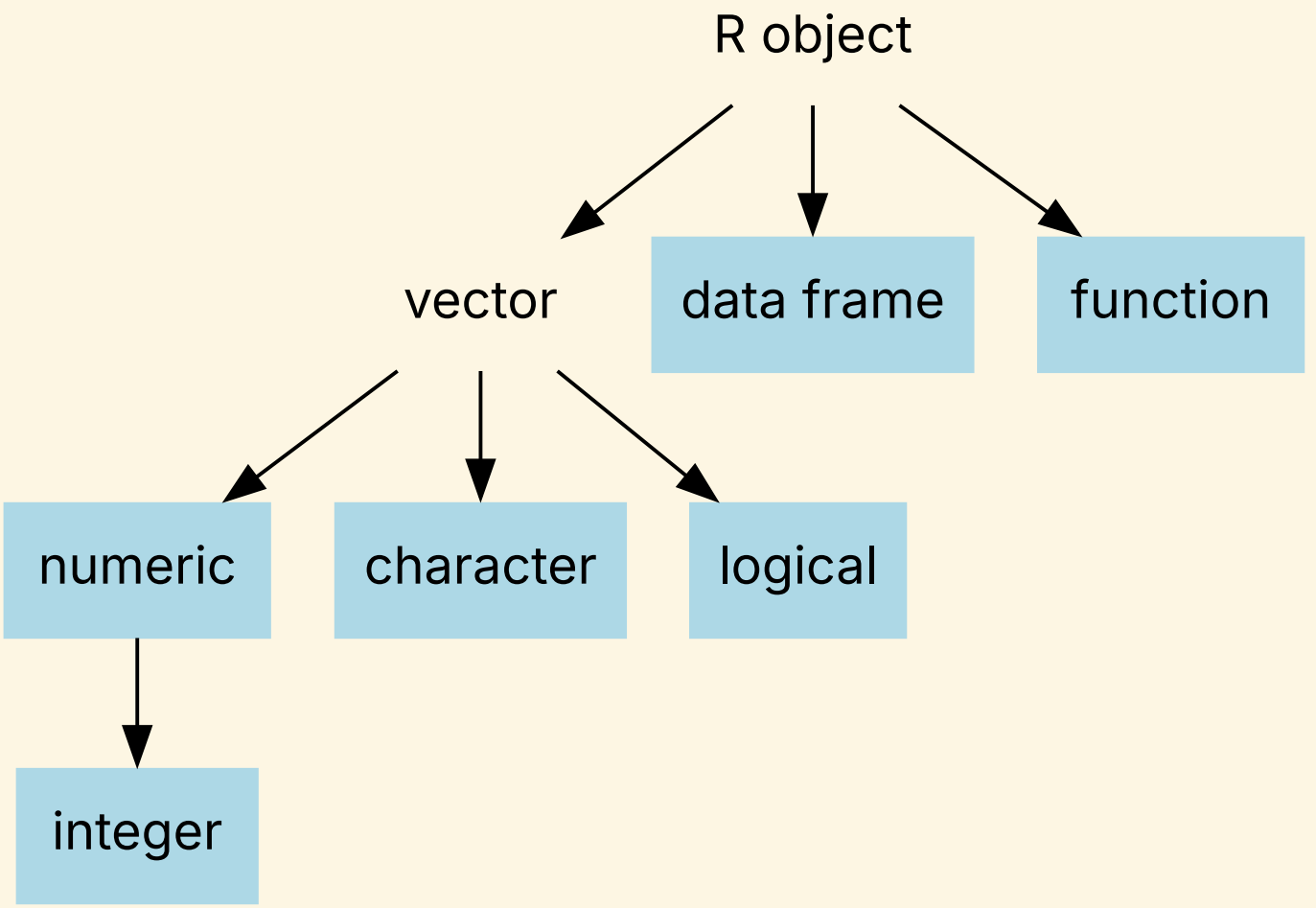
```
subset(minidata,ridageyr<59)
#|       seqn ridageyr bpxosy1 bpxodi1 hypertension
#|  1  130378       43     135      98      Stage 2
#|  3  130380       44     111      79           No
#|  4  130384       43      NA      NA         <NA>
#|  6  130386       34     110      72           No
#|  8  130388       27     130      95      Stage 2
#|  10 130390       31     113      78           No
```

Logical objects

Logical objects

```
class(minidata$ridageyr==43)
#|  [1] "logical"
class(is.na(minidata$hypertension))
#|  [1] "logical"
is.vector(minidata$ridageyr==43)
#|  [1] TRUE
is.vector(is.na(minidata$hypertension))
#|  [1] TRUE
```

Logical objects



Negation

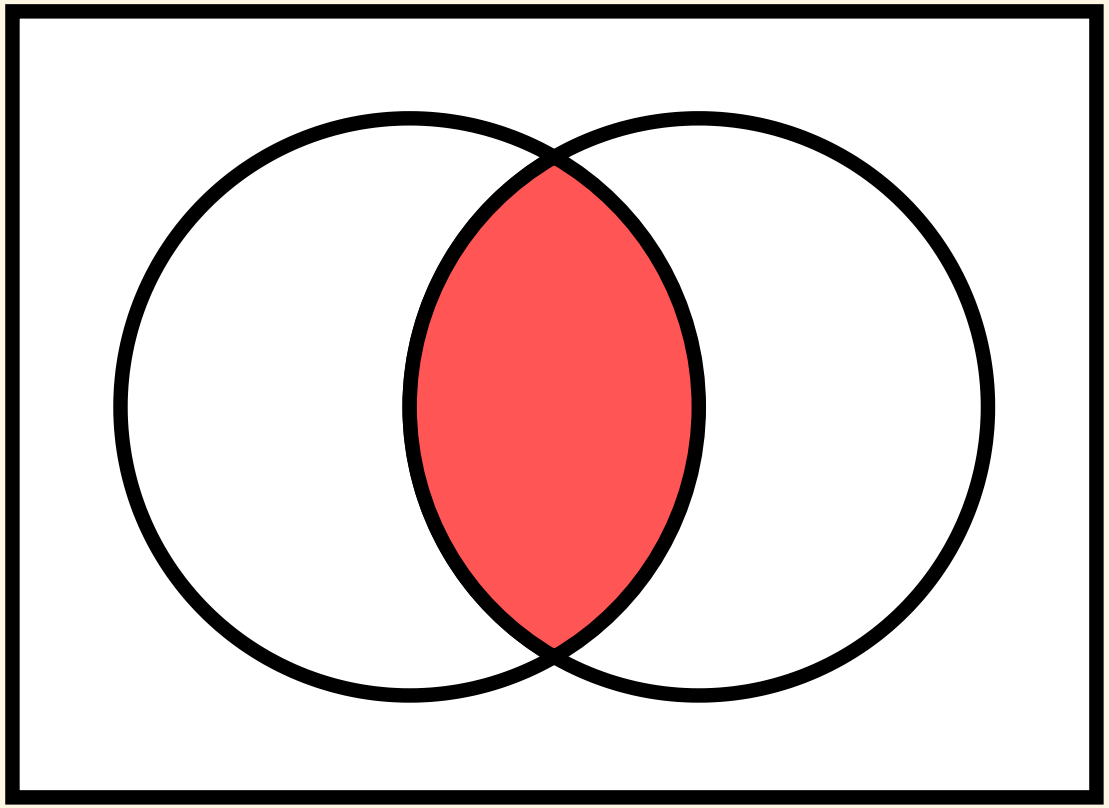
```
minidata$ridageyr==43
#|   [1]  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
minidata$ridageyr!=43
#|   [1] FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
!minidata$ridageyr==43
#|   [1] FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

- to negate (reverse) a logical object, place ! at the beginning

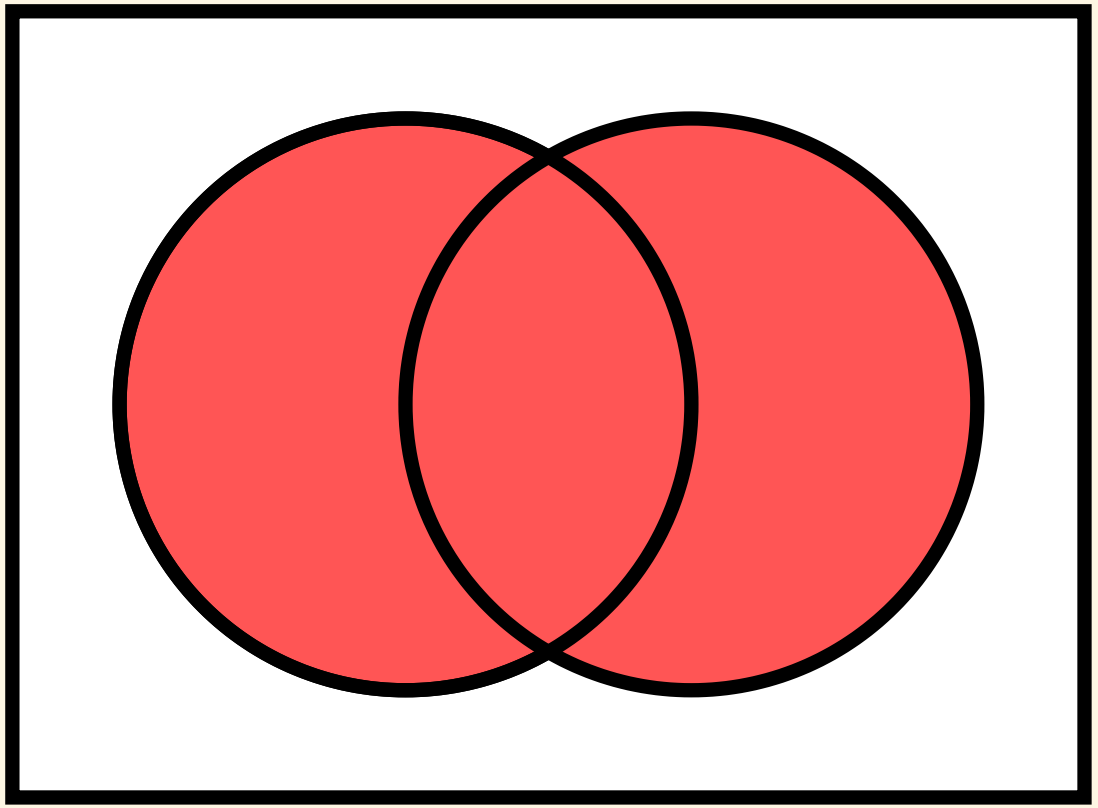
And and or

- & = and
- | = or

And



Or



And and or

- individuals without hypertension are partially defined by:

```
minidata$bpxosy1<120 & minidata$bpxodi1<80
#|   [1] FALSE FALSE  TRUE    NA    NA  TRUE FALSE FALSE FALSE  TRUE
```

- individuals with Stage 2 hypertension are defined by:

```
minidata$bpxosy1>=140 | minidata$bpxodi1>=90
#|   [1]  TRUE FALSE FALSE    NA    NA FALSE  TRUE  TRUE  TRUE FALSE
```

Logical objects and math

- when combining math with logical objects, `TRUE` values are treated as 1 and `FALSE` values are treated as 0
- this can be very useful

Logical objects and math

- to count missing values:

```
sum(is.na(minidata$hypertension))
#|  [1] 2
```

- or, to count the number of people within an age range:

```
sum(minidata$ridageyr>=30)
#|  [1] 9
```

Exercise 2

Exercise 2

1. Copy and paste the following code for defining `ex2`. Run that code.

```
ex2<-subset(nhanes,!is.na(lbxglu)&!is.na(lbdhdd))
ex2<-ex2[1:10,c('ridageyr','dmdborn4','dmdeduc2','lbxglu','lbdhdd')]
```

Exercise 2

2. Use `subset()` to get the subset of `ex2` individuals who are between 40 and 49 years of age (`ridageyr`), inclusive of both endpoints.
3. Use `subset()` to get the subset of `ex2` individuals who have a fasting glucose level (`lbxglu`) greater or equal to 100 mg/dL and an HDL cholesterol level (`lbdhdd`) greater or equal to 50 mg/dL.

Exercise 2

4. Count the number of `ex2` individuals who were born in the US (a `dmbdorn4` value equal to `1`) and have a high-school degree (a `dmdedu2` value equal to `3`).
5. Count the number of `ex2` individuals who are 50 years of age (`ridageyr`) or older and have a fasting glucose level (`lbxglu`) greater or equal to 100 mg/dL.

Exercise 2

6. Many of the functions for hypothesis testing support a `data` argument. This argument allows you to omit dollar-sign notation. So, these two examples are equivalent:

```
t.test(nhanes$bpxosy1~nhanes$asthma)
t.test(bpxosy1~asthma,data=nhanes)
```

   Modify the `data` argument in the second example to examine the relationship between systolic blood pressure and asthma among individuals with a history of smoking (a value of `smoking` equal to `History of smoking`). Remember to omit dollar-sign notation.

Exercise 2

7. Using the `data` argument, fit a simple linear regression model for the relationship between systolic blood pressure (`bpxosy1`) and diastolic blood pressure (`bpxodi1`), with systolic blood pressure as the *y* variable, restricting to individuals with a history of smoking. Remember to omit dollar-sign notation.

Defining missing values

Defining new variables

- to define a new variable, use dollar-sign notation and object assignment:

```
minidata$new<-1
```

- `new` is an arbitrary name for the new variable

Defining new variables

```
minidata
#|       seqn ridageyr bpxosy1 bpxodi1 hypertension new
#|  1  130378       43     135      98      Stage 2   1
#|  2  130379       66     121      84      Stage 1   1
#|  3  130380       44     111      79           No   1
#|  4  130384       43      NA      NA         <NA>   1
#|  5  130385       65      NA      NA         <NA>   1
#|  6  130386       34     110      72           No   1
#|  7  130387       68     143      76      Stage 2   1
#|  8  130388       27     130      95      Stage 2   1
#|  9  130389       59     145      76      Stage 2   1
#|  10 130390       31     113      78           No   1
```

Vigorous physical activity

- from NHANES, the `pad820` variable records answers to the question:

> About how long {do you/does SP} do these vigorous leisure-time physical activities each time?

Vigorous physical activity

```
table(nhanes$pad820,useNA='always')
#|
#|     1    2    3    4    5    6    7    8   10   12   15   20   25   30   32   35
#|    12    3    6    2   36    2    1    2  108    5  190  212   30  909    1   12
#|    40   45   46   50   55   60   70   75   90  100  105  120  150  180  240  300
#|    70  262    1   19    3 1064    5   13  145    2    3  364    4   93   49   21
#|   360  420  480  540  600  660  900 7777 9999 <NA>
#|    10    1    8    1    1    1    2    1   13 4466
```

Vigorous physical activity

- from NHANES:

Code or Value	Value Description	Count	Cumulative	Skip to Item
1 to 900	Range of Values	3673	3673	
7777	Refused	1	3674	
9999	Don't know	13	3687	
.	Missing	4466	8153	

Vigorous physical activity

- `7777` and `9999` are not actual minutes of vigorous physical activity
- so, we might want to recode `7777` and `9999` to be `NA`

Vigorous physical activity

- there are several possible ways to define missing values
- one approach is to use the `replace()` function:

```
nhanes$ltpa_vigorous<-replace(nhanes$pad820,
                              nhanes$pad820==7777|nhanes$pad820==9999,
                              NA)
```

Vigorous physical activity

```
nhanes$ltpa_vigorous<-replace(nhanes$pad820,
                              nhanes$pad820==7777|nhanes$pad820==9999,
                              NA)
```

- there are three arguments to `replace()` here:
  - `nhanes$pad820`: the variable that we want to modify
  - `nhanes$pad820==7777|nhanes$pad820==9999`: a rule (specifically, a logical object) that indicates which values should be replaced
  - `NA`: the replacement value

Vigorous physical activity

```
nhanes$ltpa_vigorous<-replace(nhanes$pad820,
                              nhanes$pad820==7777|nhanes$pad820==9999,
                              NA)
```

- notice that I defined a new variable, `ltpa_vigorous`

> **Tip**
>
> When cleaning variables, define new variables, rather than replacing existing variables.

Vigorous physical activity

- to check the work, use `table()`:

```
table(nhanes$ltpa_vigorous,useNA='always')
#|
#|     1    2    3    4    5    6    7    8   10   12   15   20   25   30   32   35
#|    12    3    6    2   36    2    1    2  108    5  190  212   30  909    1   12
#|    40   45   46   50   55   60   70   75   90  100  105  120  150  180  240  300
#|    70  262    1   19    3 1064    5   13  145    2    3  364    4   93   49   21
#|   360  420  480  540  600  660  900 <NA>
#|    10    1    8    1    1    1    2 4480
```

> **Tip**
>
> Always check your work after doing any data cleaning.

Exercise 3

Exercise 3

1. `pad800` stores minutes of moderate physical activity. Values of `7777` indicate "refused" while values of `9999` indicate "don't know" (the codebook is available here). Use object assignment and `replace()` to define a new variable for moderate physical activity, with values of `7777` and `9999` replaced with `NA`. Check your work.

Exercise 3

2. `alq130` stores responses to the question, "During the past 12 months, on those days that you drank alcoholic beverages, on average, how many drinks did you have?" Values of `777` indicate "refused" while values of `999` indicate "don't know." Use object assignment and `replace()` to define a new alcohol-use variable, with values of `777` and `999` replaced with `NA`. Check your work.

Exercise 3

3. Install the `baseverse` package, which we will discuss next week:

```
install.packages('baseverse')
```

4. Install the `dplyr` package, which we will discuss next week:

```
install.packages('dplyr')
```