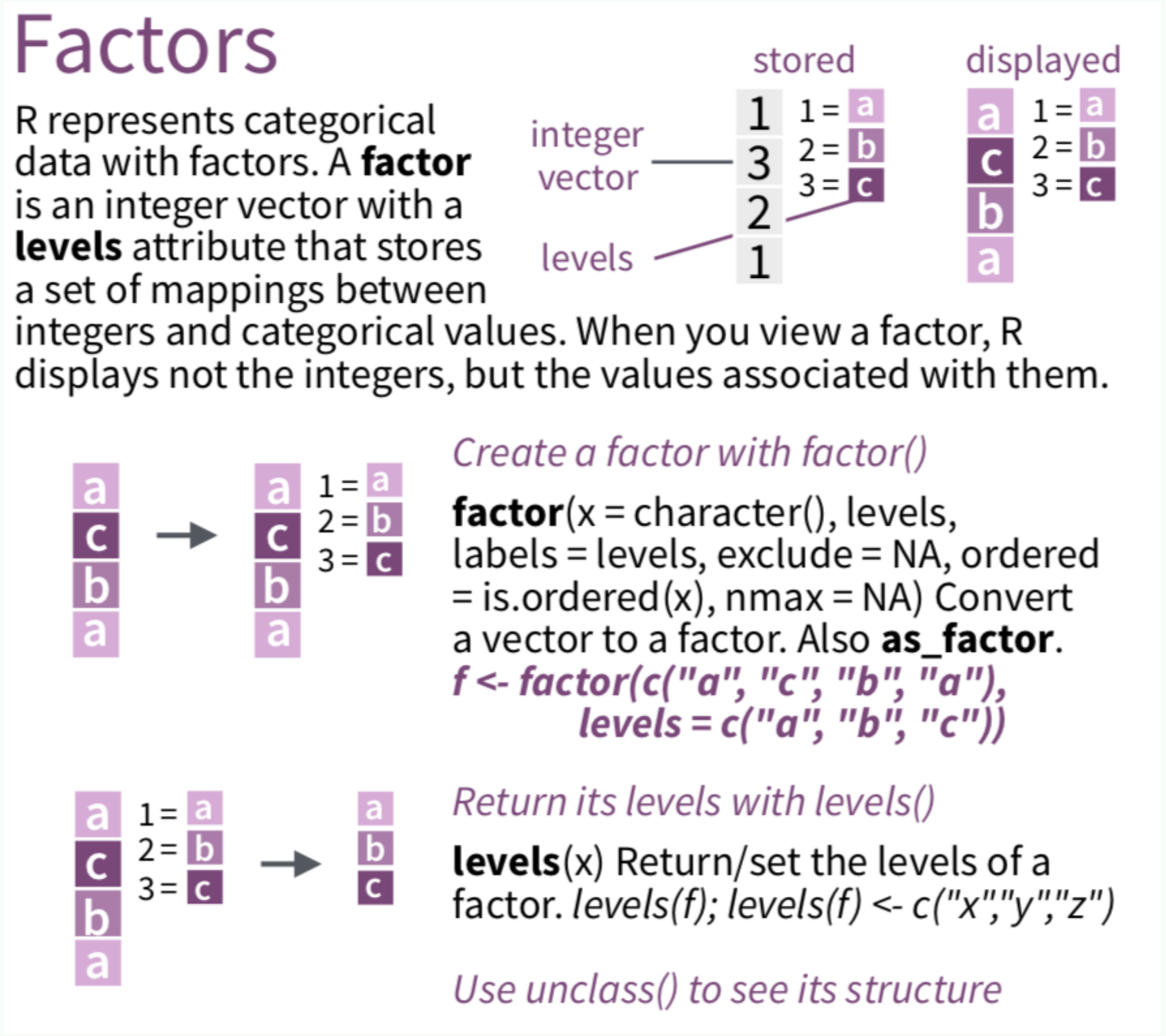
Factor manipulations

Spring 2023

April 17 2023

Factors - categorical data


stored
displayed
integer vector
levels
1 = a
2 = b
3 = c

Factors

R represents categorical data with factors. A **factor** is an integer vector with a **levels** attribute that stores a set of mappings between integers and categorical values. When you view a factor, R displays not the integers, but the values associated with them.

Create a factor with factor()

factor(x = character(), levels, labels = levels, exclude = NA, ordered = is.ordered(x), nmax = NA) Convert a vector to a factor. Also **as_factor**.
f <- factor(c("a", "c", "b", "a"), levels = c("a", "b", "c"))

Return its levels with levels()

levels(x) Return/set the levels of a factor. *levels(f); levels(f) <- c("x","y","z")*

Use unclass() to see its structure

- ***Clean and order factors with `forcats` package***
- ***Important for visualization, statistical modeling (i.e. for `lm()`), and creating tables***

See `forcats` cheatsheet and `forcats` vignette

Example - specify levels `fct_relevel()`

```
mydata <- tibble(
  id = 1:4,
  grade=c("9th","10th","11th","9th")) %>%
  mutate(grade_fac = factor(grade))
levels(mydata$grade_fac)
[1] "10th" "11th" "9th"
```

```
mydata <- mydata %>%
  mutate(
    grade_fac =
      fct_relevel(grade_fac,
                  c("9th","10th","11th")))
levels(mydata$grade_fac)
[1] "9th"  "10th" "11th"
```

```
mydata %>%
  arrange(grade_fac)
# A tibble: 4 × 3
     id grade grade_fac
  <int> <chr> <fct>
1     2 10th  10th
2     3 11th  11th
3     1 9th   9th
4     4 9th   9th
```

```
mydata %>% arrange(grade_fac)
# A tibble: 4 × 3
     id grade grade_fac
  <int> <chr> <fct>
1     1 9th   9th
2     4 9th   9th
3     2 10th  10th
4     3 11th  11th
```

Example - collapse levels `fct_collapse()`

```
mydata <- tibble(loc = c("SW","NW","NW","NE","SE","SE"))
mydata %>% mutate(
  loc_fac = factor(loc),
  loc2 = fct_collapse(loc_fac,                          # collapse levels
                      south = c("SW","SE"),
                      north = c("NE","NW")),
  loc3 = fct_lump(loc_fac,
                  n=2,
                  other_level = "other") # most common 2 levels + other
  )
# A tibble: 6 × 4
  loc   loc_fac loc2  loc3
  <chr> <fct>   <fct> <fct>
1 SW    SW      south other
2 NW    NW      north NW
3 NW    NW      north NW
4 NE    NE      north other
5 SE    SE      south SE
6 SE    SE      south SE
```

Example - collapse levels `fct_collapse()`

```
mydata <- tibble(loc = c("SW","NW","NW","NE","SE","SE"))
mydata %>% mutate(
  loc_fac = factor(loc),
  loc2 = fct_collapse(loc_fac,                          # collapse levels
                    south = c("SW","SE"),
                    north = c("NE","NW")),
  loc3 = fct_lump(loc_fac,
                n=2,
                other_level = "other") # most common 2 levels + other
  )
# A tibble: 6 × 4
  loc   loc_fac loc2  loc3
  <chr> <fct>   <fct> <fct>
1 SW    SW      south other
2 NW    NW      north NW
3 NW    NW      north NW
4 NE    NE      north other
5 SE    SE      south SE
6 SE    SE      south SE
```

Order factor levels: `fct_infreq()`

`fct_infreq()` : This function orders factor levels by their frequency in the data.

```r
# Order factor levels by their frequency
mydata <- tibble(
  id = 1:8,
  grade = c("9th", "10th", "11th", "9th", "10th", "11th", "9th", "9th")) %>%
  mutate(grade_fac = factor(grade))

mydata <- mydata %>%
  mutate(grade_fac = fct_infreq(grade_fac))

levels(mydata$grade_fac)
[1] "9th"  "10th" "11th"
```

`fct_rev()` : Reverse the order of factor levels

```r
# Reverse the order of factor levels
mydata <- tibble(
  id = 1:4,
  grade = c("9th", "10th", "11th", "9th")) %>%
  mutate(grade_fac = factor(grade))

mydata <- mydata %>%
  mutate(grade_fac = fct_rev(grade_fac))

levels(mydata$grade_fac)
[1] "9th"  "11th" "10th"
```

fct_shitf()

fct_shift(): Shift factor levels by a specified number of positions, either to the left or right.

```
# Shift factor levels to the right by 1 position
mydata <- tibble(
  id = 1:4,
  grade = c("9th", "10th", "11th", "9th")) %>%
  mutate(grade_fac = factor(grade))

mydata <- mydata %>%
  mutate(grade_fac = fct_shift(grade_fac, n = 1))

levels(mydata$grade_fac)
[1] "11th" "9th"  "10th"
```

fct_anon()

fct_anon(): Anonymize factor levels by replacing them with unique, randomly generated character strings.

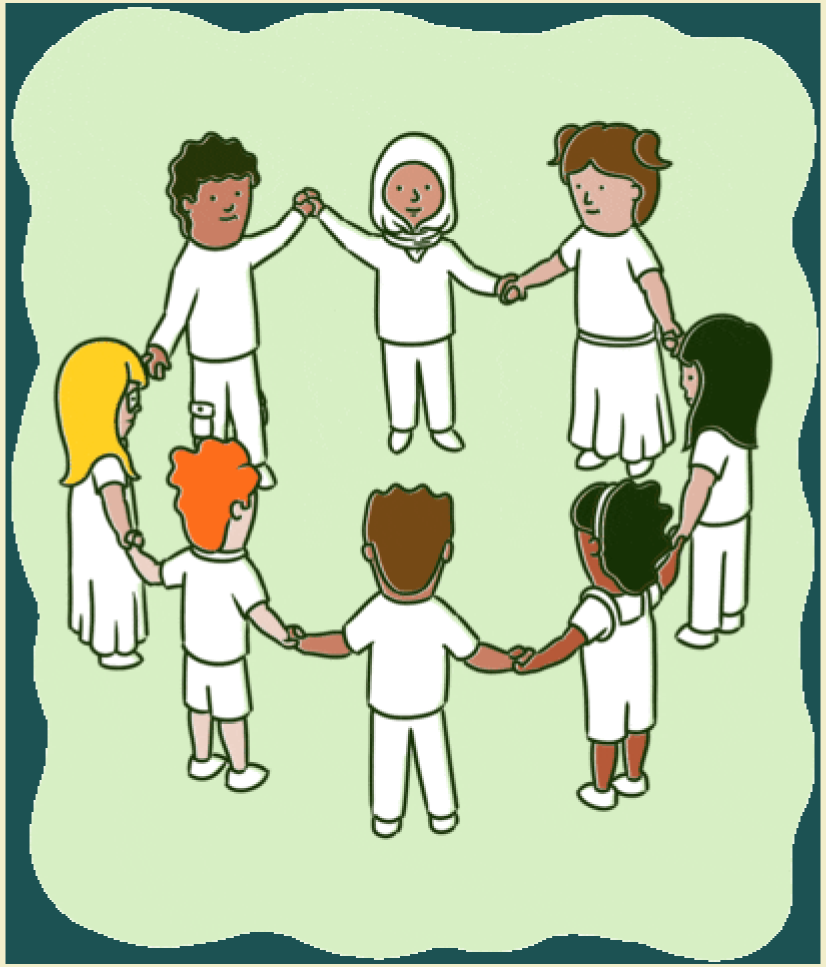
```
# Anonymize factor levels
mydata <- tibble(
  id = 1:4,
  grade = c("9th", "10th", "11th", "9th")) %>%
  mutate(grade_fac = factor(grade))

mydata <- mydata %>%
  mutate(grade_fac = fct_anon(grade_fac))

levels(mydata$grade_fac)
[1] "1" "2" "3"
```

GROUP ACTIVITY 1

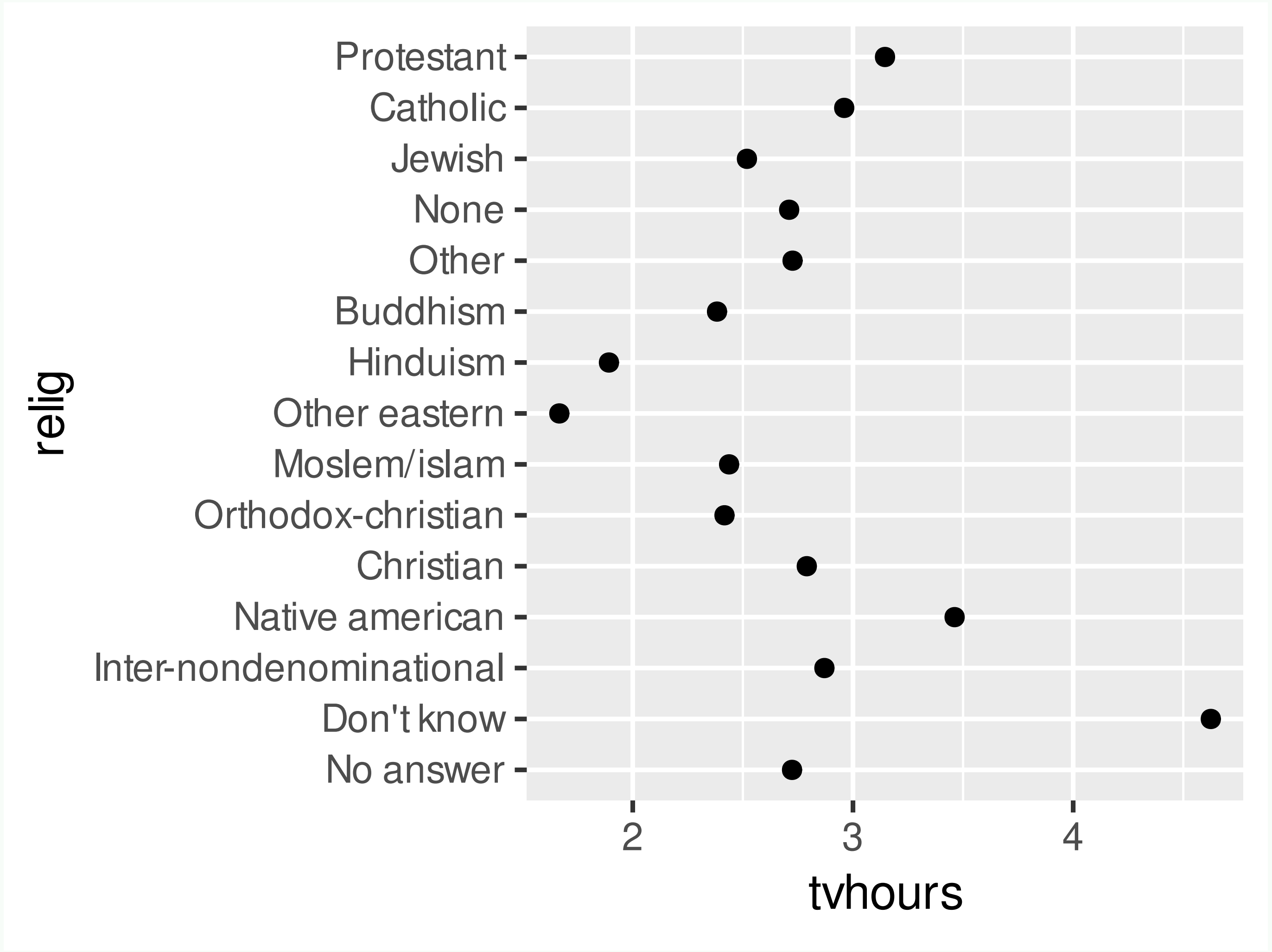
10:00



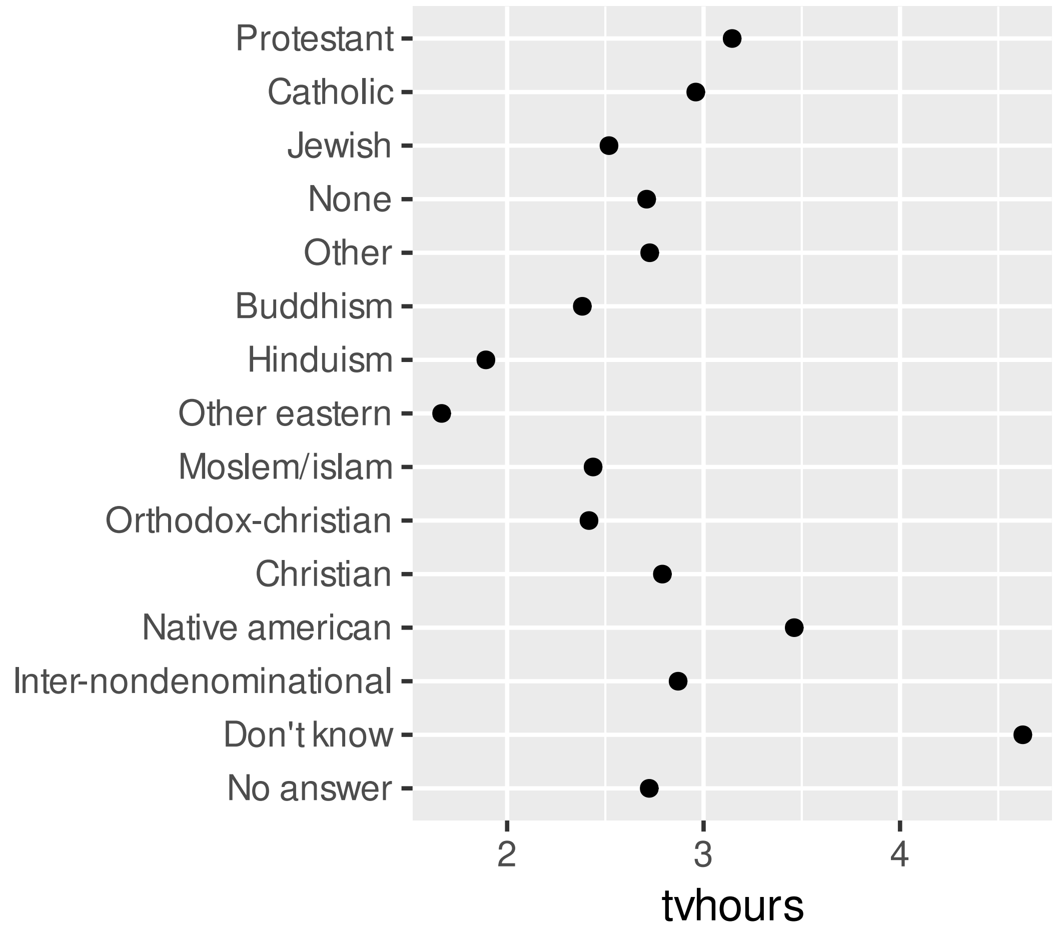
- Let's go over to maize server/ local Rstudio and our class moodle
- Get the class activity 10.Rmd file
- Work on your turn 1
- Ask me questions

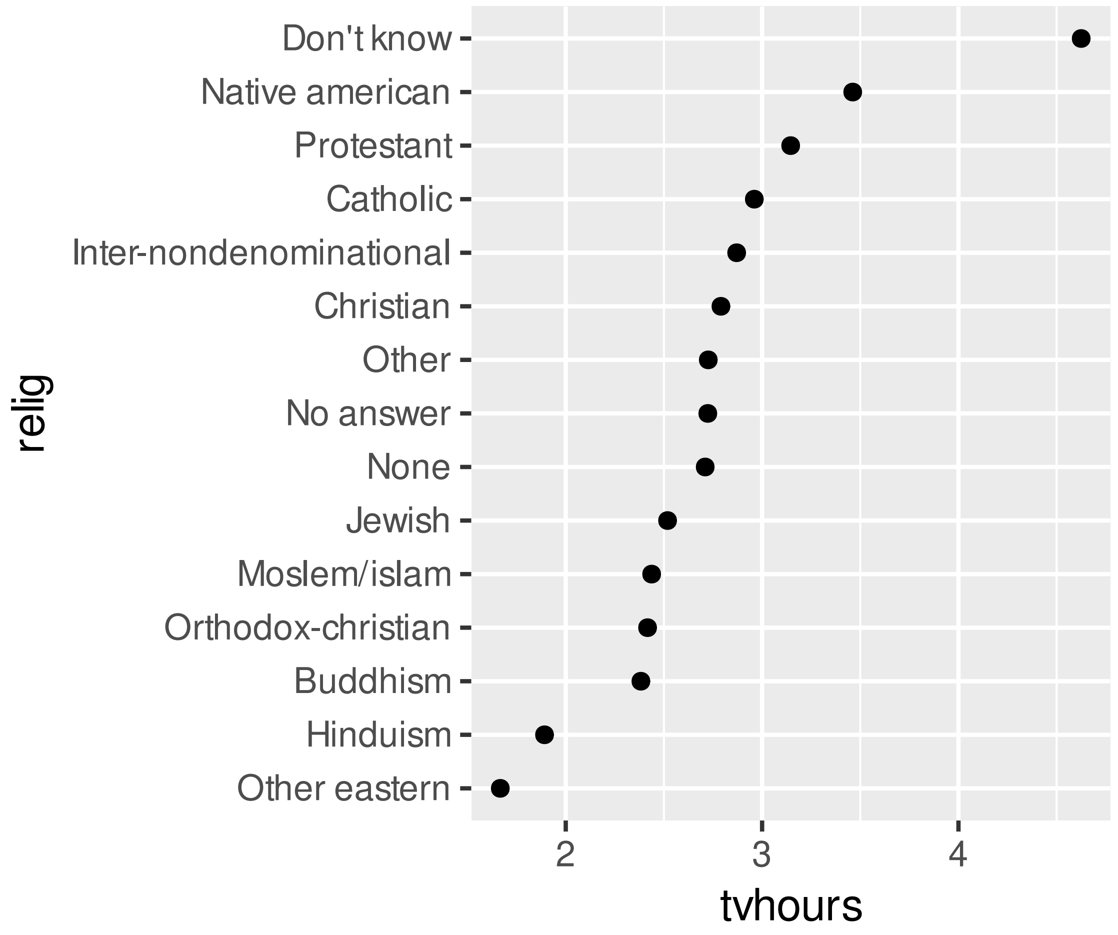
Which religions watch the least TV?

```
gss_cat %>%
  tidyr::drop_na(tvhours) %>%
  group_by(relig) %>%
  summarize(tvhours = mean(tvhours)) %>
  ggplot(aes(tvhours, relig)) +
    geom_point()
```



Which one do you prefer?





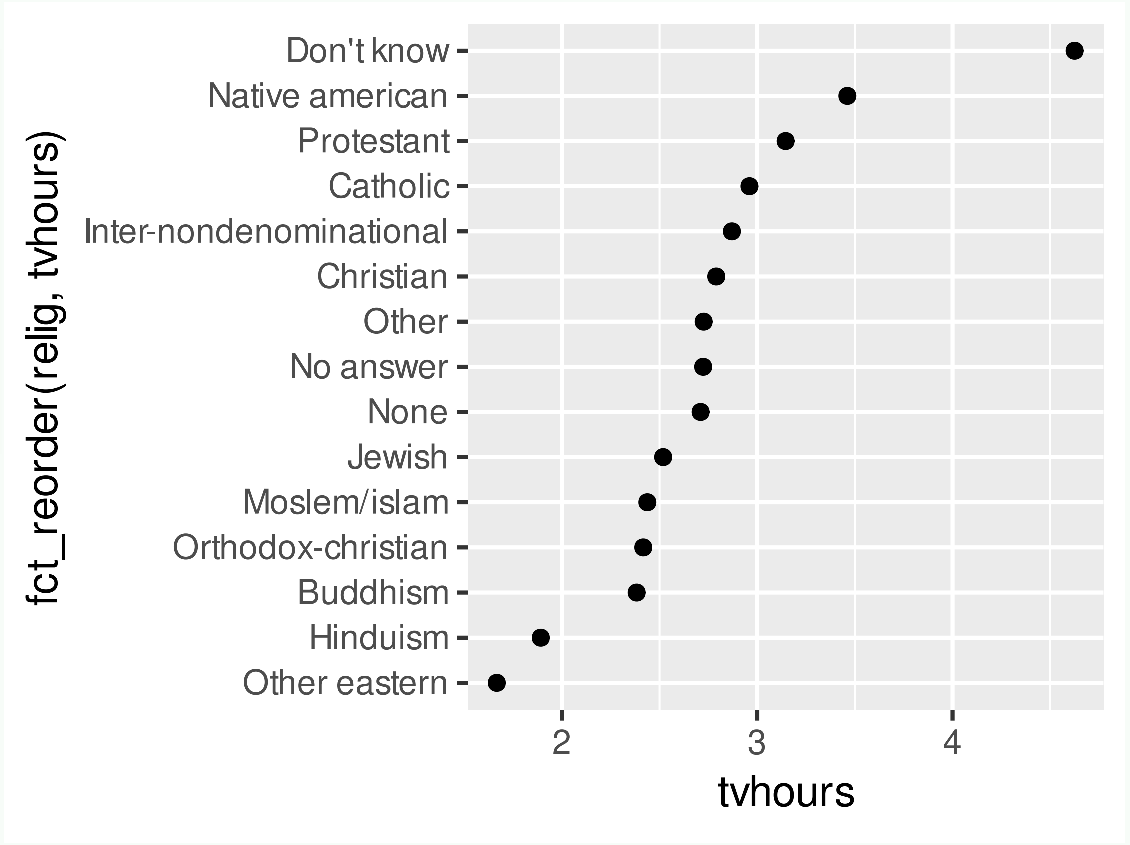
Use levels() to access a factor’s levels

```
gss_cat %>%
  pull(relig) %>%
  levels() %>%
  kable()
```

x
No answer
Don't know
Inter-nondenominational
Native american
Christian
Orthodox-christian
Moslem/islam
Other eastern
Hinduism
Buddhism
Other
None
Jewish

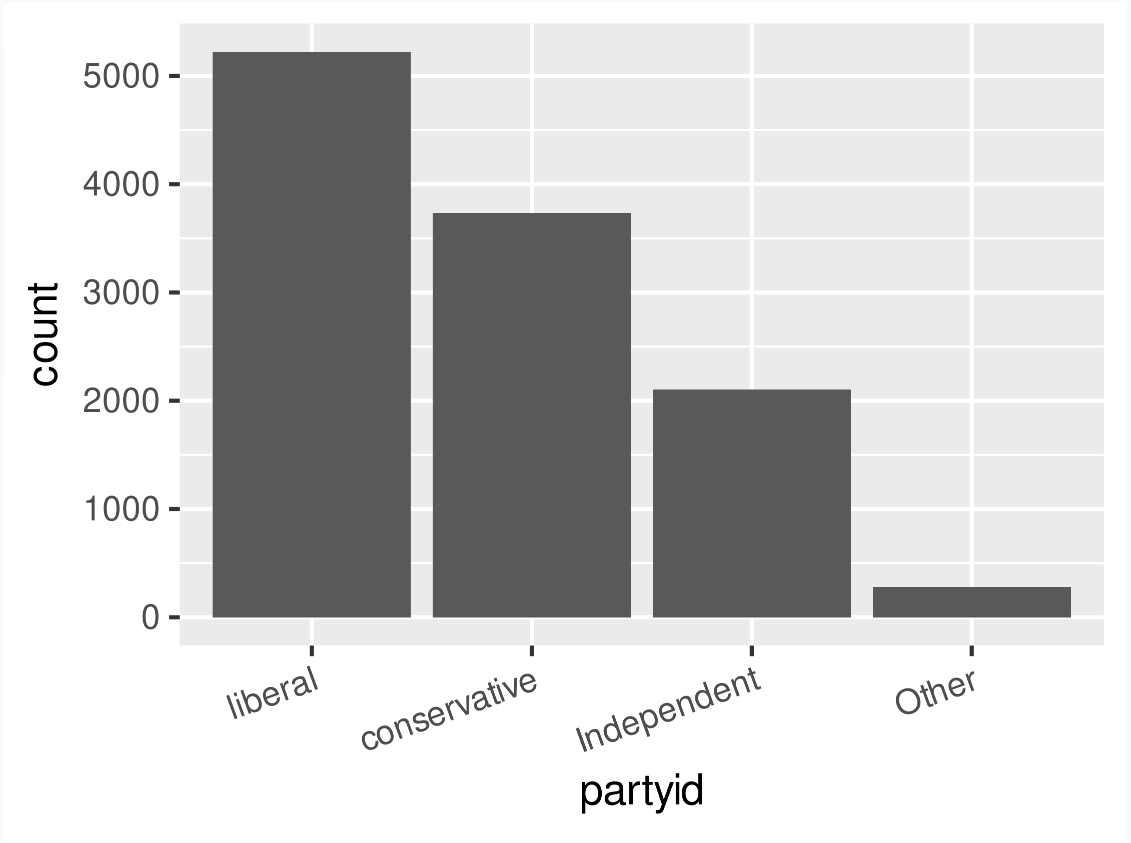
Reorder relig by tvhours

```r
gss_cat %>%
  drop_na(tvhours) %>%
  group_by(relig) %>%
  summarize(tvhours = mean(tvhours)) %>%
  ggplot(aes(
    x = tvhours,
    y = fct_reorder(relig, tvhours)
  )) +
    geom_point()
```



Lumping partyid: fct_lump()

```
gss_cat %>%
  mutate(partyid = fct_lump(partyid, n = 3)) %>%
  ggplot(aes(x = fct_infreq(partyid))) +
  geom_bar() +
  theme(axis.text.x = element_text(angle = 20,
                                   vjust = 1,
                                   hjust=1)) +
  labs(x = "partyid")
```



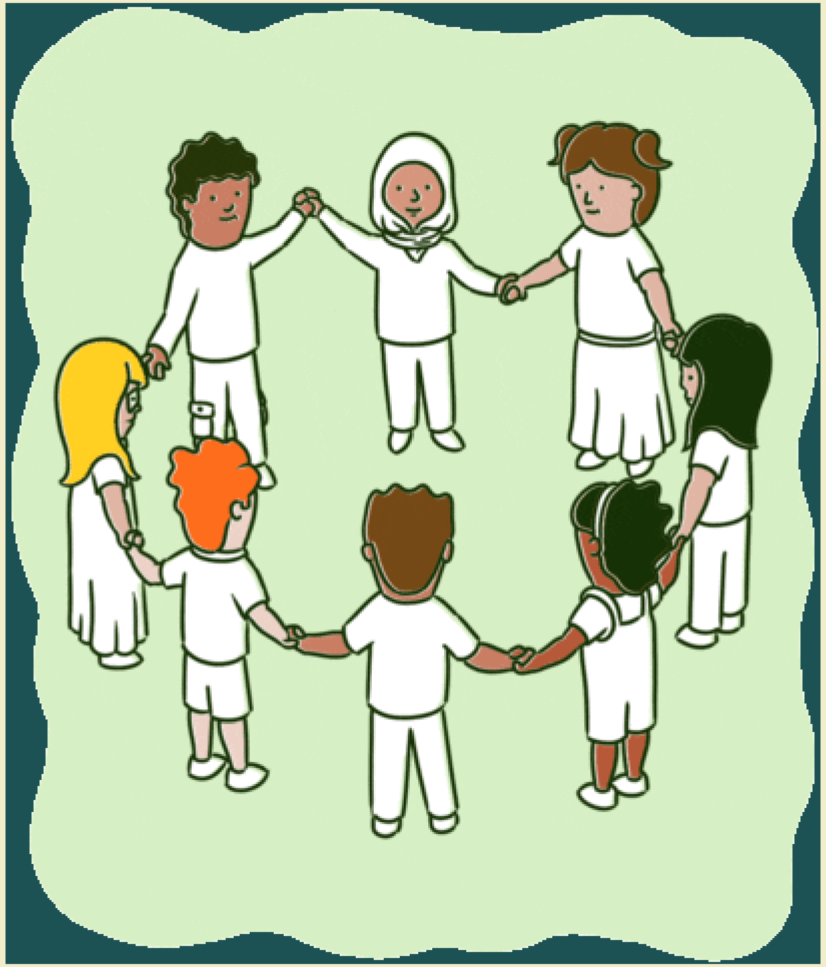
Summary

To enhance your data analysis, you can use the following factor manipulation techniques:

- ***Reorder the levels to arrange them in a meaningful order.***
- ***Recode the levels to modify the labels or merge similar categories.***
- ***Collapse levels to group multiple categories into one.***
- ***Lump levels to reduce the number of categories by combining less frequent ones.***

GROUP ACTIVITY 2

10:00



- *Work on your turn 2*
- *Ask me questions*