IMMERSE day 3: asynchronous practice

Casey O'Hara

2023-05-26

```r
### Set message and warning to FALSE to suppress warnings and messages during
### package loading (e.g, library(tidyverse))
knitr::opts_chunk$set(echo = TRUE, message = FALSE, warning = FALSE)
```

```r
library(poLCA) ### LCA package with some more data we'll use
library(tidyverse)
  ### dialect of R - cleans up and standardizes a lot of wrangling and data vis.
  ### note the packages listed in the message!

library(here) ### helps with creating relative pathways relative to .Rproj
```

Let's apply ideas from Day 3 with some new data

Now let's try some wrangling with some data on cheating in high school, built into the `poLCA` package (polytomous latent class analysis).

```r
data(cheating) ### load sample data built into the poLCA package

### examine the data in a few ways: summary(), ggplot() - type these in console

# summary(cheating)
# glimpse(cheating)
# ?cheating
```

Some things to note:

- the variable names are all shouting! use `janitor::clean_names()` to tone them down (and eliminate spaces, punctuation, etc) - defaults to snake case.
- the values are all double precision floating point numbers, except GPA which is integer
- all except GPA are scored as 1 (FALSE) or 2 (TRUE)
- GPA is scored as 1 (GPA < 3), 2 (3.00-3.25), 3 (3.26-3.50), 4 (3.51-3.75) and 5 (3.76-4.00).

```r
cheating <- cheating %>%
  janitor::clean_names()

cheating_sum <- cheating %>%
  group_by(gpa) %>%
  summarize(mean_lieexam = mean(lieexam),
            mean_liepaper = mean(liepaper),
            mean_fraud = mean(fraud),
            mean_copyexam = mean(copyexam))
cheating_sum
```

```
## # A tibble: 6 x 5
##     gpa mean_lieexam mean_liepaper mean_fraud mean_copyexam
##   <int>        <dbl>         <dbl>      <dbl>         <dbl>
## 1     1         1.18          1.18       1.08          1.33
## 2     2         1.12          1.12       1.08          1.21
## 3     3         1.04          1.10       1.04          1.15
## 4     4         1             1.06       1.03          1.12
## 5     5         1.03          1.03       1.07          1.07
## 6    NA         1             1          1             1
```

- What does the `cheating_sum` result mean?
  - for each GPA value, calculate the mean of `lieexam`, `liepaper`, etc; a mean of 1 means all `FALSE`, a mean of 2 means all `TRUE`, a mean of 1.25 means mostly `FALSE` etc.
- Note: leaving column values as numbers, they will probably not be accurately interpreted as categorical!

- How should we recode the `lieexam`, `liepaper`, `fraud`, and `copyexam` variables? (`T/F`, `"true"/"false"`, `"lie"/"no lie"`, 1/0, . . .) Consider pros and cons, then recode according to your preference (or mix-and-match!)
- How should we recode the GPA? multi-value discrete, or multiple columns of binary? Consider pros and cons, then recode according to your preference.
- Examples of each of these options shown below!

Working with factors in R

So far, we're not explicitly working with factors, though the categorical values in the penguins dataframe are built-in as factors. Let's use the `poLCA::cheaters` dataframe to work with factors a bit more.

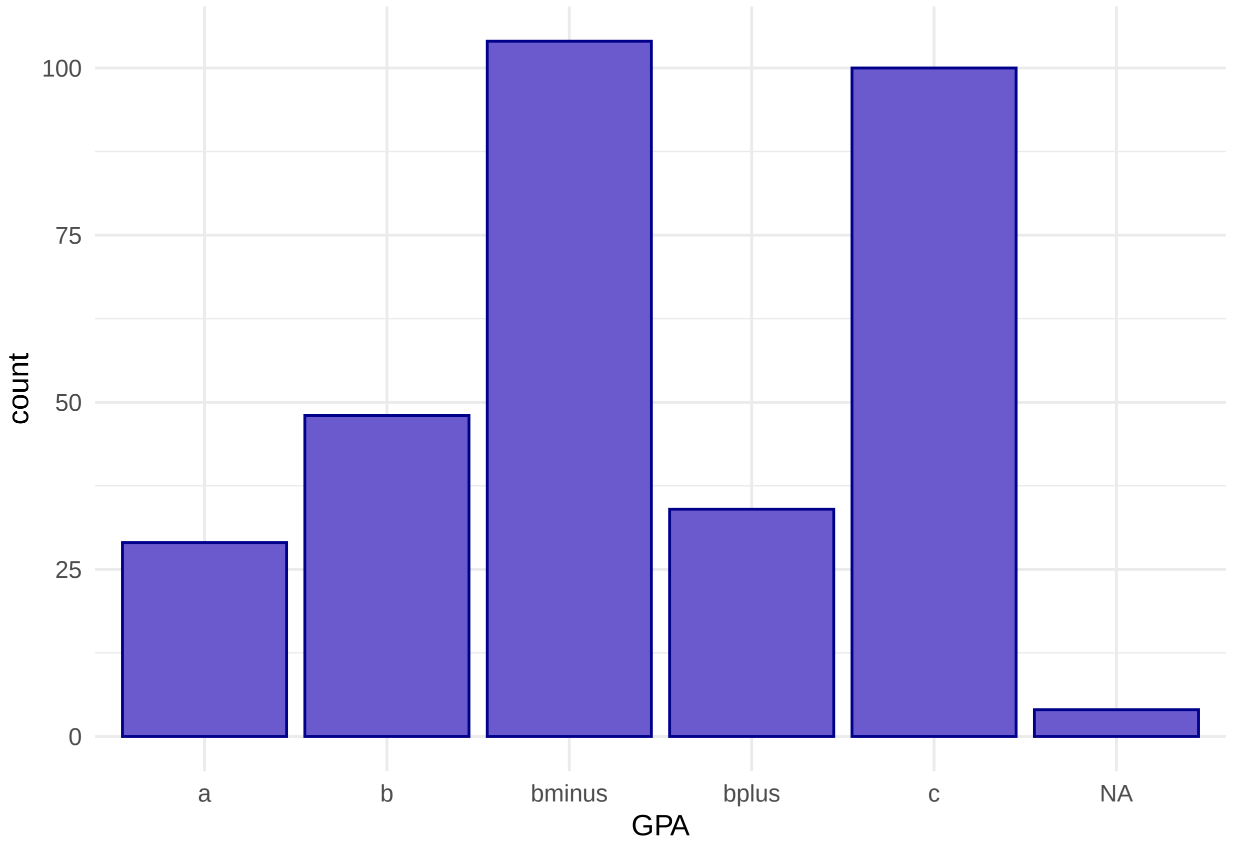
Let's recode binary to TRUE/FALSE, recode GPA to `'c', 'bminus', 'b', 'bplus', 'a'`.

```
gpa_lvls <- c('c', 'bminus', 'b', 'bplus', 'a') ### leave out missing?

cheat_recode <- cheating %>%
  mutate(lieexam_lgl  = ifelse(lieexam == 1, FALSE, TRUE),
         liepaper_lgl = (liepaper == 2),
         fraud_lgl    = (fraud == 2),
         copyexam_lgl = case_when(copyexam == 1 ~ FALSE,
                                  copyexam == 2 ~ TRUE,
                                  TRUE          ~ NA) ) %>%
  mutate(gpa_cat = case_when(gpa == 1 ~ 'c',
                             gpa == 2 ~ 'bminus',
                             gpa == 3 ~ 'b',
                             gpa == 4 ~ 'bplus',
                             gpa == 5 ~ 'a',
                             TRUE     ~ NA_character_)) #%>%
# mutate(gpa_cat = factor(gpa_cat, levels = gpa_lvls))
```

Let's ggplot a bar chart of grades - note anything weird?

```r
ggplot(cheat_recode) +
  geom_bar(aes(x = gpa_cat), fill = 'slateblue', color = 'darkblue') +
  theme_minimal() +
  labs(x = 'GPA')
```



```r
### note grades not in appropriate order (instead, alphabetical order)!
### turn into a factor!
```

More methods for `mutate`

See `across()` for applying `mutate` across multiple columns simultaneously! The first mutate here (line 114) does the same as lines 80-85 above, by identifying which columns are class `double`, then applying a function of "does the value in this column (`.x`) match the number 2?"

Also we can take advantage of indexes to assign values from a vector. Line 115 says, create a column called gpa_chr (for "GPA in `character` type"), and to each row, use the numeric value of `gpa` to select which element of the gpa_lvls vector. E.g., if `gpa` is 2, select the second value of `gpa_lvls` (i.e., `'bminus'`).

```r
cheat_recode2 <- cheating %>%
  mutate(across(.cols = where(is.double), .fns = ~(.x == 2), .names = '{.col}_lgl')) %>%
  mutate(gpa_chr = gpa_lvls[gpa])
```

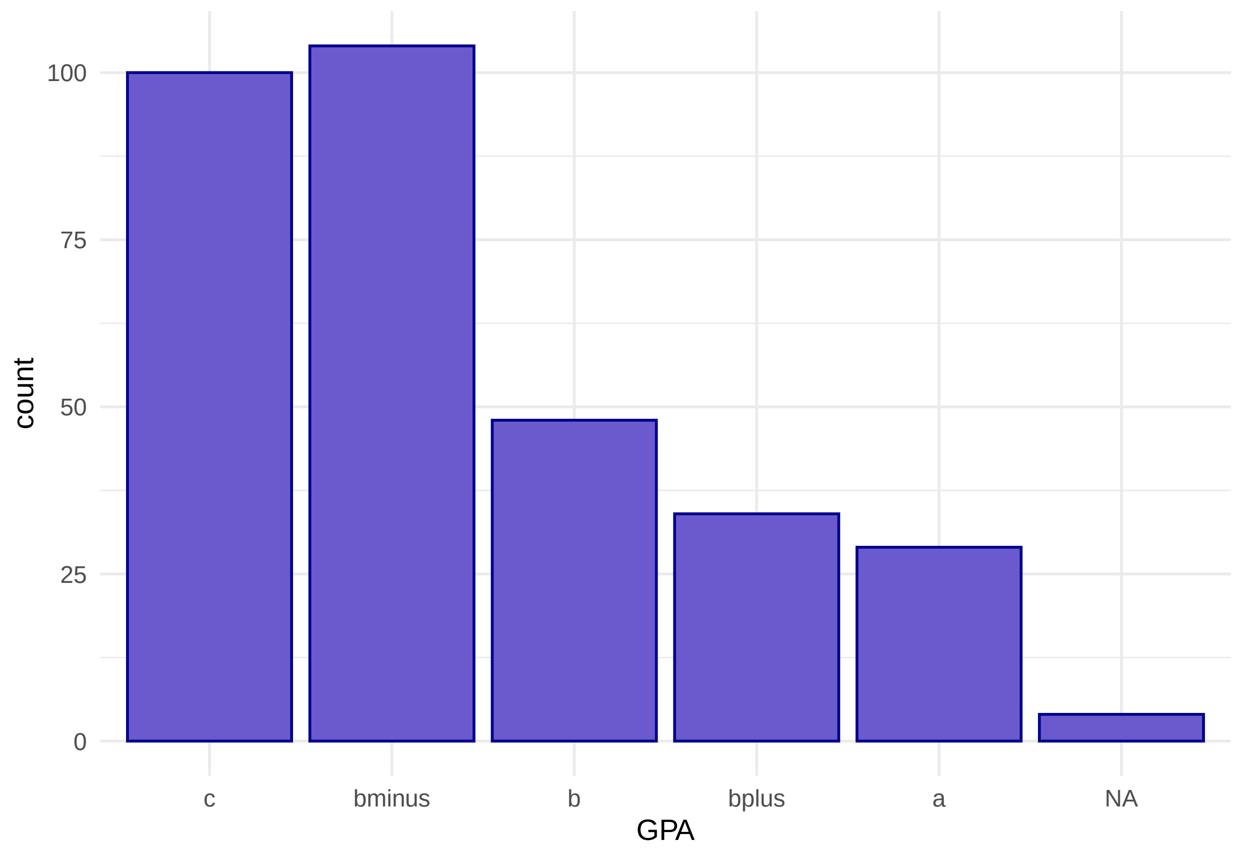
Using factors, we can turn the `gpa_cat` column from `character` to a `factor` and control the order on the plot, but ALSO the order in which a model function (e.g., `lm()` and probably LCA and MPlus modeling functions) interprets the levels!

```r
glimpse(cheat_recode2)
```

```
## Rows: 319
## Columns: 10
## $ lieexam      <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ liepaper     <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ fraud        <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ copyexam     <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ gpa          <int> NA, NA, NA, NA, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ lieexam_lgl  <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F~
## $ liepaper_lgl <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F~
## $ fraud_lgl    <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F~
## $ copyexam_lgl <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, F~
## $ gpa_chr      <chr> NA, NA, NA, NA, "c", "c", "c", "c", "c", "c", "c", "c", "~
```

```r
cheat_fct <- cheat_recode2 %>%
  mutate(gpa_fct = factor(gpa_chr, levels = gpa_lvls)) %>%
  mutate(gpa_ord = factor(gpa_chr, levels = rev(gpa_lvls), ordered = TRUE)) %>%
  mutate(gpa_fct2 = forcats::fct_reorder(gpa_chr, gpa))

ggplot(cheat_fct) +
  ### try replacing gpa_fct with gpa_ord here - what changes?
  geom_bar(aes(x = gpa_fct), fill = 'slateblue', color = 'darkblue') +
  theme_minimal() +
  labs(x = 'GPA')
```



```
### try these in the console:
# glimpse(cheat_fct)
# head(cheat_fct$gpa_chr)
# head(cheat_fct$gpa_fct)
# head(cheat_fct$gpa_ord)
```

Tidy up and write out a cleaned/prepped dataset

Let's clean up one version of our cheating datasets with logical and factor values, then write it out for later use, e.g., if we were to write another script that will ingest the data for modeling. Note, depending on the order in which we loaded the packages, we might get an error with `select()` so we can force R to use the `dplyr::select()`!

```
cheat_fct_clean <- cheat_fct %>%
  # select(gpa, gpa_fct, lieexam_cat, ends_with('_lgl')) ### error! check ?select
  dplyr::select(gpa, gpa_fct, lieexam = lieexam_lgl, ends_with('_lgl')) %>%
    ### or load tidyverse after poLCA!
  rename(liepaper = liepaper_lgl, fraud = fraud_lgl, copyexam = copyexam_lgl)

head(cheat_fct_clean)
```

```
##   gpa gpa_fct lieexam liepaper fraud copyexam
## 1  NA    <NA>   FALSE    FALSE FALSE    FALSE
## 2  NA    <NA>   FALSE    FALSE FALSE    FALSE
## 3  NA    <NA>   FALSE    FALSE FALSE    FALSE
## 4  NA    <NA>   FALSE    FALSE FALSE    FALSE
## 5   1       c   FALSE    FALSE FALSE    FALSE
## 6   1       c   FALSE    FALSE FALSE    FALSE
```

```
### create a data folder, then save out our clean dataframe
write_csv(cheat_fct_clean, here('data', 'cheating_data_cleaned.csv'))
```

You may need to create a `data` folder in your R project to make this work!

Now read in the cleaned data - what do you note about our factor column? it's just a character column again! R doesn't store the "factorness" of the column in the saved file. Consider how to turn it back into a factor again, with levels in the right order? Try it and see what you can figure out, BEFORE looking at the code below.

```
cheat_clean <- read_csv(here('data/cheating_data_cleaned.csv'))
glimpse(cheat_clean)
```

```
## Rows: 319
## Columns: 6
## $ gpa      <dbl> NA, NA, NA, NA, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ gpa_fct  <chr> NA, NA, NA, NA, "c", "c", "c", "c", "c", "c", "c", "c", "c", ~
## $ lieexam  <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE~
## $ liepaper <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE~
## $ fraud    <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE~
## $ copyexam <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE~
```

```
cheat_clean <- cheat_clean %>%
  mutate(gpa_fct = fct_reorder(gpa_fct, gpa))
glimpse(cheat_clean)
```

```
## Rows: 319
## Columns: 6
## $ gpa      <dbl> NA, NA, NA, NA, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ~
## $ gpa_fct  <fct> NA, NA, NA, NA, c, c, c, c, c, c, c, c, c, c, c, c, c, c, c, ~
## $ lieexam  <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE~
## $ liepaper <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE~
## $ fraud    <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE~
## $ copyexam <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE~
```

```
head(cheat_clean$gpa_fct)
```

```
## [1] <NA> <NA> <NA> <NA> c    c
## Levels: c bminus b bplus a
```