Problem Set #4

Nicole Wood (Team NJ with Jade Jevandofsky and Jessika Viveros

October 21, 2023

In last week's problem set (Week 3, Question 5), you used `tidyverse` functions such as `filter()`, `arrange()`, and `select()` to perform data manipulations. In this problem set, you'll be asked to incorporate the use of pipes (`%>%`) into your code.

You'll also be practicing variable creation in this problem set, using both `mutate()` in combination with `if_else()`, `case_when()`, and `recode()`.

Question 1: Data manipulation using pipes

1. In the code chunk below, complete the following:

   - Load the `tidyverse` library
   - Use the `load()` and `url()` functions to download the `df_school_all` dataframe from the url: `https://github.com/anyone-can-cook/rclass1/raw/master/data/recruiting/recruit_school_allvars.Rd`
     - Each row in `df_school_all` represents a high school (includes both public and private)
     - There are columns (e.g., `visit_by_100751`) indicating the number of times a university visited that high school
     - The variable `total_visits` identifies the number of visits the high school received from all (16) public research universities in this data collection sample

```
r = getOption("repos")
r["CRAN"] = "http://cran.us.r-project.org"
options(repos = r)
rm(list = ls())
install.packages("tidyverse")
#> Installing package into 'C:/Users/there/AppData/Local/R/win-library/4.3'
#> (as 'lib' is unspecified)
#> package 'tidyverse' successfully unpacked and MD5 sums checked
#>
#> The downloaded binary packages are in
#>  C:\Users\there\AppData\Local\Temp\RtmpiW6WAk\downloaded_packages
library(tidyverse)
#> -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
#> v dplyr     1.1.3     v readr     2.1.4
#> v forcats   1.0.0     v stringr   1.5.0
#> v ggplot2   3.4.4     v tibble    3.2.1
#> v lubridate 1.9.3     v tidyr     1.3.0
#> v purrr     1.0.2
#> -- Conflicts ------------------------------------------ tidyverse_conflicts() --
#> x dplyr::filter() masks stats::filter()
```

```
#> x dplyr::lag()    masks stats::lag()
#> i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error

load(url("https://github.com/anyone-can-cook/rclass1/raw/master/data/recruiting/recruit_school_allvars.
```

2. Use the functions `arrange()`, `select()`, and `head()` to do the following:

   - Sort `df_school_all` descending by `total_visits`
   - Select the following variables: `name`, `state_code`, `city`, `school_type`, `total_visits`, `med_inc`, `pct_white`, `pct_black`, `pct_hispanic`, `pct_asian`, `pct_amerindian`
   - Show the first 10 rows of the dataframe, which represents the top 10 most visited schools by the 16 universities

   Complete this in 2 ways: (1) without using pipes (the exact same way you did last week) and (2) using pipes (`%>%`) to complete this task using 1 line of code

Without using pipes:

```
df_school_all_2 <- arrange(df_school_all, desc(total_visits))
df_school_all_2 <- select(df_school_all_2, name, state_code, city, school_type, total_visits, med_inc, p
head(df_school_all_2, n = 10)
#> # A tibble: 10 x 11
#>    name    state_code city  school_type total_visits med_inc pct_white pct_black
#>    <chr>   <chr>      <chr> <chr>              <int>   <dbl>     <dbl>     <dbl>
#>  1 EPISCO~ VA         ALEX~ private               26 109558.     77.8      12.1
#>  2 Lyons ~ IL         La G~ public                23  94306.     74.1       3.71
#>  3 ALLEN ~ TX         ALLEN public                23 100809      57.2      11.8
#>  4 COPPEL~ TX         COPP~ public                23 123382.     49.9       4.97
#>  5 FLOWER~ TX         FLOW~ public                22 157234.     74         3.06
#>  6 NOLAN ~ TX         FORT~ private               21  39490.     55.8       3.47
#>  7 FORT W~ TX         FORT~ private               20  89470.      4.09      2.82
#>  8 LOVEJO~ TX         LUCAS public                19 100809      81.9       1.91
#>  9 STRAKE~ TX         HOUS~ private               18  29630.     56.7       7.76
#> 10 TRINIT~ TX         ADDI~ private               18  77380      83.5       1.60
#> # i 3 more variables: pct_hispanic <dbl>, pct_asian <dbl>, pct_amerindian <dbl>
```

Using pipes ('%>%'):

```
df_school_all_3 <- arrange(df_school_all, desc(total_visits)) %>% select(name, state_code, city, school_
df_school_all_3 %>% head(n=10)
#> # A tibble: 10 x 11
#>    name    state_code city  school_type total_visits med_inc pct_white pct_black
#>    <chr>   <chr>      <chr> <chr>              <int>   <dbl>     <dbl>     <dbl>
#>  1 EPISCO~ VA         ALEX~ private               26 109558.     77.8      12.1
#>  2 Lyons ~ IL         La G~ public                23  94306.     74.1       3.71
#>  3 ALLEN ~ TX         ALLEN public                23 100809      57.2      11.8
#>  4 COPPEL~ TX         COPP~ public                23 123382.     49.9       4.97
#>  5 FLOWER~ TX         FLOW~ public                22 157234.     74         3.06
#>  6 NOLAN ~ TX         FORT~ private               21  39490.     55.8       3.47
#>  7 FORT W~ TX         FORT~ private               20  89470.      4.09      2.82
#>  8 LOVEJO~ TX         LUCAS public                19 100809      81.9       1.91
#>  9 STRAKE~ TX         HOUS~ private               18  29630.     56.7       7.76
#> 10 TRINIT~ TX         ADDI~ private               18  77380      83.5       1.60
#> # i 3 more variables: pct_hispanic <dbl>, pct_asian <dbl>, pct_amerindian <dbl>
```

3. Building upon the previous question, use the functions `arrange()`, `select()`, `filter()`, and `head()` to do the following (select same variables as above):

   (A) Top 10 most visited public high schools in California
   (B) Top 10 most visited private high schools in California

   Complete this in 2 ways: (1) without using pipes (the exact same way you did last week) and (2) using pipes (%>%) to complete the tasks using 1 line of code each

Without using pipes:

```
#Top 10 most visited public high schools in California

df_school_all_public <- arrange(df_school_all, desc(total_visits))
df_school_all_public <- select(df_school_all_public, name, state_code, city, school_type, total_visits,
df_school_all_public <- filter(df_school_all_public, school_type == "public", state_code == "CA")
head(df_school_all_public, n = 10)
#> # A tibble: 10 x 11
#>    name    state_code city  school_type total_visits med_inc pct_white pct_black
#>    <chr>   <chr>      <chr> <chr>              <int>   <dbl>     <dbl>     <dbl>
#>  1 Corona~ CA         Newp~ public                12 133966       82.6     0.900
#>  2 Trabuc~ CA         Miss~ public                12 112446.      57.2     1.69
#>  3 Monte ~ CA         Danv~ public                10 168605       67.9     0.931
#>  4 Santa ~ CA         Sant~ public                10  93942       41.4     9.28
#>  5 Tustin~ CA         Tust~ public                10  70780.      13.3     3.26
#>  6 Calaba~ CA         Cala~ public                 9 123449       78.7     4.17
#>  7 Palos ~ CA         Palo~ public                 9 211304.      69.5     2.28
#>  8 Mira C~ CA         Manh~ public                 8 168271       58.8     5.36
#>  9 Burrou~ CA         Burb~ public                 8  87288       37.2     2.34
#> 10 Aliso ~ CA         Alis~ public                 8 110660.      59.2     1.28
#> # i 3 more variables: pct_hispanic <dbl>, pct_asian <dbl>, pct_amerindian <dbl>

#Top 10 most visited private high schools in California

df_school_all_private <- arrange(df_school_all, desc(total_visits))
df_school_all_private <- select(df_school_all_private, name, state_code, city, school_type, total_visits
df_school_all_private <- filter(df_school_all_private, school_type == "private", state_code == "CA")
head(df_school_all_private, n = 10)
#> # A tibble: 10 x 11
#>    name    state_code city  school_type total_visits med_inc pct_white pct_black
#>    <chr>   <chr>      <chr> <chr>              <int>   <dbl>     <dbl>     <dbl>
#>  1 SANTA ~ CA         RANC~ private               15 105576.      66.6      1.27
#>  2 JSERRA~ CA         SAN ~ private               14  88324       60.1      1.93
#>  3 MATER ~ CA         SANT~ private               12  64052.      38.3      3.72
#>  4 SERVIT~ CA         ANAH~ private               11  55142       41.0      4.53
#>  5 ST FRA~ CA         LA C~ private                9 177146.      48.0      1.66
#>  6 CHAMIN~ CA         WEST~ private                8  64568.      49.1      6.77
#>  7 NOTRE ~ CA         SHER~ private                8  91428.      62.6      7.40
#>  8 JUNIPE~ CA         SAN ~ private                8 123328       61.7      2.97
#>  9 CATHED~ CA         SAN ~ private                8 143160       87.1      2.12
#> 10 ST IGN~ CA         SAN ~ private                6 121018.      60.1      3.18
#> # i 3 more variables: pct_hispanic <dbl>, pct_asian <dbl>, pct_amerindian <dbl>
```

Using pipes ('%>%'):

```
#Top 10 most visited public high schools in California

df_school_all_public2 <- arrange(df_school_all, desc(total_visits)) %>% select(name, state_code, city, s
df_school_all_public2 %>% head(n=10)
#> # A tibble: 10 x 11
#>    name    state_code city  school_type total_visits med_inc pct_white pct_black
#>    <chr>   <chr>      <chr> <chr>              <int>   <dbl>     <dbl>     <dbl>
#>  1 Corona~ CA         Newp~ public                12 133966       82.6     0.900
#>  2 Trabuc~ CA         Miss~ public                12 112446.      57.2     1.69
#>  3 Monte ~ CA         Danv~ public                10 168605       67.9     0.931
#>  4 Santa ~ CA         Sant~ public                10  93942       41.4     9.28
#>  5 Tustin~ CA         Tust~ public                10  70780.      13.3     3.26
#>  6 Calaba~ CA         Cala~ public                 9 123449       78.7     4.17
#>  7 Palos ~ CA         Palo~ public                 9 211304.      69.5     2.28
#>  8 Mira C~ CA         Manh~ public                 8 168271       58.8     5.36
#>  9 Burrou~ CA         Burb~ public                 8  87288       37.2     2.34
#> 10 Aliso ~ CA         Alis~ public                 8 110660.      59.2     1.28
#> # i 3 more variables: pct_hispanic <dbl>, pct_asian <dbl>, pct_amerindian <dbl>

#Top 10 most visited private high schools in California

df_school_all_private2 <- arrange(df_school_all, desc(total_visits)) %>% select(name, state_code, city,
df_school_all_private2 %>% head(n=10)
#> # A tibble: 10 x 11
#>    name    state_code city  school_type total_visits med_inc pct_white pct_black
#>    <chr>   <chr>      <chr> <chr>              <int>   <dbl>     <dbl>     <dbl>
#>  1 SANTA ~ CA         RANC~ private               15 105576.      66.6      1.27
#>  2 JSERRA~ CA         SAN ~ private               14  88324       60.1      1.93
#>  3 MATER ~ CA         SANT~ private               12  64052.      38.3      3.72
#>  4 SERVIT~ CA         ANAH~ private               11  55142       41.0      4.53
#>  5 ST FRA~ CA         LA C~ private                9 177146.      48.0      1.66
#>  6 CHAMIN~ CA         WEST~ private                8  64568.      49.1      6.77
#>  7 NOTRE ~ CA         SHER~ private                8  91428.      62.6      7.40
#>  8 JUNIPE~ CA         SAN ~ private                8 123328       61.7      2.97
#>  9 CATHED~ CA         SAN ~ private                8 143160       87.1      2.12
#> 10 ST IGN~ CA         SAN ~ private                6 121018.      60.1      3.18
#> # i 3 more variables: pct_hispanic <dbl>, pct_asian <dbl>, pct_amerindian <dbl>
```

Question 2: Variable creation using `tidyverse`'s `mutate()`

Often before creating new "analysis" variables, you may want to investigate the values of "input" variables. Here are some examples of checking variable values using `count()`:

```
# Counts the total number of observations (i.e., rows) in `df_school_all`
df_school_all %>% count()

# Counts the number of observations that have missing values for the variable `med_inc`
df_school_all %>% filter(is.na(med_inc)) %>% count()

# Frequency count of the variable `school_type`
df_school_all %>% count(school_type)
```

1. Use `mutate()` with `if_else()` to create a 0/1 indicator and then use `count()` to generate the following frequency tables:

    - Create 0/1 indicator called `ca_school` for whether the high school is in California and generate the frequency table for the values of `ca_school`
    - Create 0/1 indicator called `ca_pub_school` for whether the high school is a public school in California and generate the frequency table for the values of `ca_pub_school`

    Note: You do not need to assign/retain the indicator variables in the `df_school_all` dataframe.

```
df_school_all %>%
  mutate(ca_school = if_else(state_code == "CA", 1, 0)) %>%
    count(ca_school)
#> # A tibble: 2 x 2
#>   ca_school     n
#>       <dbl> <int>
#> 1         0 19531
#> 2         1  1770

df_school_all %>%
  mutate(ca_pub_school = if_else(state_code == "CA" & school_type == "public", 1, 0)) %>%
    count(ca_pub_school)
#> # A tibble: 2 x 2
#>   ca_pub_school     n
#>           <dbl> <int>
#> 1             0 19897
#> 2             1  1404
```

2. Complete the following steps to create an analysis variable using `mutate()` and `if_else()`:

    - First, use `select()` to select the variables `name`, `pct_black`, `pct_hispanic`, `pct_amerindian` from `df_school_all`, and assign the resulting dataframe to `df_race`. You'll be using `df_race` for the remaining bullet points below.
    - Use `filter()`, `is.na()`, and `count()` to investigate whether or not the following variables have missing values: `pct_black`, `pct_hispanic`, `pct_amerindian`
    - Use `mutate()` to create a new variable `pct_bl_hisp_nat` in `df_race` that is the sum of `pct_black`, `pct_hispanic`, and `pct_amerindian`. Remember to assign to `df_race`.
    - Create a 0/1 indicator called `gt50pct_bl_hisp_nat` for whether more than 50% of students identify as black, latinx, or native american and generate a frequency table for the values of `gt50pct_bl_hisp_nat`

```
df_race <- select(df_school_all, name, pct_black, pct_hispanic, pct_amerindian)

#lines 155 - 162 were written to verify that the total na count is actually 0.

df_race %>% filter(is.na(pct_black)) %>%
  count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0

df_race %>% filter(is.na(pct_hispanic)) %>%
```

```
  count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0

df_race %>% filter(is.na(pct_amerindian)) %>%
  count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0

df_race %>% filter(is.na(pct_black) | is.na(pct_hispanic) | is.na(pct_amerindian)) %>%
  count()
#> # A tibble: 1 x 1
#>       n
#>   <int>
#> 1     0

df_race <- mutate(df_race, pct_bl_hisp_nat = pct_black + pct_hispanic + pct_amerindian)

df_race <- mutate(df_race, gt50pct_bl_hisp_nat = if_else(pct_bl_hisp_nat > 50, 1, 0))

df_race %>% count(gt50pct_bl_hisp_nat)
#> # A tibble: 2 x 2
#>   gt50pct_bl_hisp_nat     n
#>                 <dbl> <int>
#> 1                   0 15701
#> 2                   1  5600
```

3. Complete the following steps to create an analysis variable using `mutate()` and `case_when()`:

   - First, use `select()` to select the variables `name` and `state_code` from `df_school_all`, and assign the resulting dataframe to `df_schools`
   - Use `case_when()` to create a new variable in `df_schools` called `region` whose values are:
     - `Northeast`, if `state_code` is in: `'CT', 'ME', 'MA', 'NH', 'RI', 'VT', 'NJ', 'NY', 'PA'`
     - `Midwest`, if `state_code` is in: `'IN', 'IL', 'MI', 'OH', 'WI', 'IA', 'KS', 'MN', 'MO', 'NE', 'ND', 'SD'`
     - `West`, if `state_code` is in: `'AZ', 'CO', 'ID', 'NM', 'MT', 'UT', 'NV', 'WY', 'AK', 'CA', 'HI', 'OR', 'WA'`
     - `South`, if `state_code` is not any of the above states (Hint: Use `TRUE` as the condition to specify default value. You can see an example here.)

```
df_schools <- select(df_school_all, name, state_code)

df_schools <- mutate(df_schools, region = case_when(
  state_code %in% c("CT", "ME", "MA", "NH", "RI", "VT", "NJ", "NY", "PA") ~ "Northeast",
  state_code %in% c("IN", "IL", "MI", "OH", "WI", "IA", "KA", "MN", "MO", "NE", "ND", "SD") ~ "Midwest"
  state_code %in% c("AZ", "CO", "ID", "NM", "MT", "UT", "NV", "WY", "AK", "CA", "HI", "OR", "WA") ~ "We
  .default = "South"))
```

```r
head(df_schools, n = 10)
#> # A tibble: 10 x 3
#>    name                        state_code region
#>    <chr>                       <chr>      <chr>
#>  1 Bethel Regional High School AK         West
#>  2 Ayagina'ar Elitnaurvik      AK         West
#>  3 Kwigillingok School         AK         West
#>  4 Nelson Island Area School   AK         West
#>  5 Alakanuk School             AK         West
#>  6 Emmonak School              AK         West
#>  7 Hooper Bay School           AK         West
#>  8 Ignatius Beans School       AK         West
#>  9 Pilot Station School        AK         West
#> 10 Kotlik School              AK         West
tail(df_schools, n = 10)
#> # A tibble: 10 x 3
#>    name                                    state_code region
#>    <chr>                                   <chr>      <chr>
#>  1 Tongue River High School                WY         West
#>  2 Sheridan High School                    WY         West
#>  3 Shoshoni High School                    WY         West
#>  4 Green River High School                 WY         West
#>  5 Jackson Hole High School                WY         West
#>  6 Upton High School                       WY         West
#>  7 Worland High School                     WY         West
#>  8 Saint Stephen's Indian School           WY         West
#>  9 JOURNEYS SCHOOL OF TETON SCIENCE SCHOOLS WY         West
#> 10 JACKSON HOLE COMMUNITY SCHOOL           WY         West
```

4. Complete the following steps to recode variables using `mutate()` and `recode()`:

   - In the `df_schools` dataframe, replace the values of the `region` variable as follows:
     - Change `Northeast` to `NE`
     - Change `Midwest` to `MW`
     - Change `West` to `W`
     - Change `South` to `S`
   - In the `df_schools` dataframe, create a new variable `state_name` whose value is:
     - `California`, if `state_code` is `CA`
     - `New York`, if `state_code` is `NY`
     - Choose another state of your choice to recode
     - `Other`, if `state_code` is any other state (Hint: Use `.default` to specify the default value)

```r
names(df_schools)
#> [1] "name"       "state_code" "region"

df_schools <- mutate(df_schools, region = recode(region,
  "Northeast" = "NE",
  "Midwest" = "MW",
  "West" = "W",
  "South" = "S"))

head(df_schools)
```

```
#> # A tibble: 6 x 3
#>   name                          state_code region
#>   <chr>                         <chr>      <chr>
#> 1 Bethel Regional High School AK         W
#> 2 Ayagina'ar Elitnaurvik      AK         W
#> 3 Kwigillingok School         AK         W
#> 4 Nelson Island Area School   AK         W
#> 5 Alakanuk School             AK         W
#> 6 Emmonak School              AK         W

df_schools <- mutate(df_schools, state_name = recode(state_code,
  "CA" = "California",
  "NY" = "New York",
  "TN" = "Tennessee",
  .default = "Other"))

head(df_schools)
#> # A tibble: 6 x 4
#>   name                          state_code region state_name
#>   <chr>                         <chr>      <chr>  <chr>
#> 1 Bethel Regional High School AK         W      Other
#> 2 Ayagina'ar Elitnaurvik      AK         W      Other
#> 3 Kwigillingok School         AK         W      Other
#> 4 Nelson Island Area School   AK         W      Other
#> 5 Alakanuk School             AK         W      Other
#> 6 Emmonak School              AK         W      Other
```

Create a GitHub issue

- Go to the class repository and create a new issue.
- Refer to rclass1 student issues readme for instructions on how to post questions or reflections.
- You are also required to respond to at least one issue posted by another student.
- Paste the url to your issue here: https://github.com/anyone-can-cook/rclass1_student_issues_f23/issues/432
- Paste the url to the issue you responded to here: https://github.com/anyone-can-cook/rclass1_student_issues_f23/issues/522

Knit to pdf and submit problem set

Knit to pdf by clicking the "Knit" button near the top of your RStudio window (icon with blue yarn ball) or drop down and select "Knit to PDF"

- Go to the class website and under the "Readings & Assignments" » "Week 4" tab, click on the "Problem set 4 submission link"
- Submit both .Rmd and pdf files

- Use this naming convention "lastname_firstname_ps#" for your .Rmd and pdf files (e.g. jaquette_ozan_ps4.Rmd & jaquette_ozan_ps4.pdf)