PS630 Lab6 Notes

Haohan Chen

October 19, 2018

Load the data

```
load("lab6_data_exercise.Rdata")
# Describe the dataset
str(data_country)
```

```
## 'data.frame':    86 obs. of  5 variables:
##  $ country_code    : chr  "AGO" "ALB" "ARG" "AUS" ...
##  $ country_name    : chr  "Angola" "Albania" "Argentina" "Australia" ...
##  $ polity2_score   : num  -2 9 8 10 10 8 7 -6 9 8 ...
##  $ GDP_pcap_ppp    : num  5085 6707 12502 32735 35537 ...
##  $ country_category: chr  "Africa" "Post Communist" "Latin America" "Advanced Capitalist" ...
```

```
str(data_party)
```

```
## 'data.frame':    495 obs. of  7 variables:
##  $ country_code  : chr  "ALB" "ALB" "ALB" "ALB" ...
##  $ country_name  : chr  "Albania" "Albania" "Albania" "Albania" ...
##  $ party_name    : chr  "UHRP" "DPA" "SPA" "RPA" ...
##  $ local_presence: num  2.12 1 1 2 1.27 ...
##  $ client_effort : num  13 14.3 14.3 12.7 13.7 ...
##  $ left_right    : num  4.67 5.5 4 6.89 3.89 ...
##  $ partysize     : num  1.3 24.6 40.5 18.5 4.1 ...
```

The package

```
library(dplyr)
library(tidyr)
library(ggplot2)

# Convert data frmes to tibbles
data_country <- as_tibble(data_country)
data_party <- as_tibble(data_party)
```

Review of Labs 4 and 5

0. What does the `%>%` command do?
1. Create a new dataset from `data_country`, with only variables `country_code` and `GDP_pcap_ppp`
2. With the above dataset, create a variable `log_GDP_pcap_ppp` that is the logarithm of `GDP_pcap_ppp`
3. With the above data, create a new dataset from `data_country`, with only observations (countries) that are in the `Advanced Capitalist` category (variable `country_category`)
4. With the above data, sort observations by `GDP_pcap_ppp`, (1) from high to low; (2) from low to high
5. With the above data (or data of 3), find top 5 countries in terms of GDP per capita

```r
# Your code here
# 1 (select)
data_country %>% select(country_code, GDP_pcap_ppp)
```

```
## # A tibble: 86 x 2
##    country_code GDP_pcap_ppp
##    <chr>               <dbl>
##  1 AGO                 5085.
##  2 ALB                 6707.
##  3 ARG                12502.
##  4 AUS                32735.
##  5 AUT                35536.
##  6 BEL                33399.
##  7 BEN                 1239.
##  8 BGD                 1172.
##  9 BGR                10529.
## 10 BOL                 3972.
## # ... with 76 more rows
```

```r
data_country2 <- data_country %>% select(country_code, GDP_pcap_ppp)

# 2 (mutate)
# 2. With the above dataset, create a variable `log_GDP_pcap_ppp` that is the logarithm of `GDP_pcap_pp
data_country %>%
  mutate(log_GDP_pcap_ppp = log(GDP_pcap_ppp))
```

```
## # A tibble: 86 x 6
##    country_code country_name polity2_score GDP_pcap_ppp country_category
##    <chr>        <chr>                <dbl>        <dbl> <chr>
##  1 AGO          Angola                  -2        5085. Africa
##  2 ALB          Albania                  9        6707. Post Communist
##  3 ARG          Argentina                8       12502. Latin America
##  4 AUS          Australia               10       32735. Advanced Capitali~
##  5 AUT          Austria                 10       35536. Advanced Capitali~
##  6 BEL          Belgium                  8       33399. Advanced Capitali~
##  7 BEN          Benin                    7        1239. Africa
##  8 BGD          Bangladesh              -6        1172. Asia/Mideast
##  9 BGR          Bulgaria                 9       10529. Post Communist
## 10 BOL          Bolivia                  8        3972. Latin America
## # ... with 76 more rows, and 1 more variable: log_GDP_pcap_ppp <dbl>
```

```r
mutate(data_country, log_GDP_pcap_ppp = log(GDP_pcap_ppp))
```

```
## # A tibble: 86 x 6
##    country_code country_name polity2_score GDP_pcap_ppp country_category
##    <chr>        <chr>                <dbl>        <dbl> <chr>
##  1 AGO          Angola                  -2        5085. Africa
##  2 ALB          Albania                  9        6707. Post Communist
##  3 ARG          Argentina                8       12502. Latin America
##  4 AUS          Australia               10       32735. Advanced Capitali~
##  5 AUT          Austria                 10       35536. Advanced Capitali~
##  6 BEL          Belgium                  8       33399. Advanced Capitali~
##  7 BEN          Benin                    7        1239. Africa
##  8 BGD          Bangladesh              -6        1172. Asia/Mideast
##  9 BGR          Bulgaria                 9       10529. Post Communist
```

```
## 10 BOL          Bolivia                   8        3972. Latin America
## # ... with 76 more rows, and 1 more variable: log_GDP_pcap_ppp <dbl>
```

```
# 3 (filter)
# 3. With the above data, create a new dataset from `data_country`, with only observations (countries)

data_country %>%
  mutate(log_GDP_pcap_ppp = log(GDP_pcap_ppp)) %>%
  filter(country_category == "Advanced Capitalist"
         & polity2_score > 9)
```

```
## # A tibble: 16 x 6
##    country_code country_name polity2_score GDP_pcap_ppp country_category
##    <chr>        <chr>                <dbl>        <dbl> <chr>
##  1 AUS          Australia               10       32735. Advanced Capitali~
##  2 AUT          Austria                 10       35536. Advanced Capitali~
##  3 CHE          Switzerland             10       37581. Advanced Capitali~
##  4 DEU          Germany                 10       33181. Advanced Capitali~
##  5 DNK          Denmark                 10       34905. Advanced Capitali~
##  6 ESP          Spain                   10       28536. Advanced Capitali~
##  7 FIN          Finland                 10       33324. Advanced Capitali~
##  8 GBR          UK                      10       33717. Advanced Capitali~
##  9 GRC          Greece                  10       26928. Advanced Capitali~
## 10 IRL          Ireland                 10       41036. Advanced Capitali~
## 11 ITA          Italy                   10       28682. Advanced Capitali~
## 12 NLD          Netherlands             10       36956. Advanced Capitali~
## 13 NOR          Norway                  10       49359. Advanced Capitali~
## 14 NZL          New Zealand             10       25281. Advanced Capitali~
## 15 PRT          Portugal                10       21169. Advanced Capitali~
## 16 SWE          Sweden                  10       34090. Advanced Capitali~
## # ... with 1 more variable: log_GDP_pcap_ppp <dbl>
```

```
# 4 (arrange)
#4. With the above data, sort observations by `GDP_pcap_ppp`, (1) from high to low; (2) from low to hig

data_country %>% arrange(wt = GDP_pcap_ppp)
```

```
## # A tibble: 86 x 5
##    country_code country_name polity2_score GDP_pcap_ppp country_category
##    <chr>        <chr>                <dbl>        <dbl> <chr>
##  1 NER          Niger                    6         592. Africa
##  2 MOZ          Mozambique               6         758. Africa
##  3 MLI          Mali                     6        1023. Africa
##  4 TZA          Tanzania                 1        1141. Africa
##  5 BGD          Bangladesh              -6        1172. Asia/Mideast
##  6 BEN          Benin                    7        1239. Africa
##  7 GHA          Ghana                    8        1260. Africa
##  8 ZMB          Zambia                   5        1283. Africa
##  9 KEN          Kenya                    7        1456. Africa
## 10 SEN          Senegal                  8        1573. Africa
## # ... with 76 more rows
```

```
data_country %>% arrange(wt = -GDP_pcap_ppp)
```

```
## # A tibble: 86 x 5
##    country_code country_name polity2_score GDP_pcap_ppp country_category
```

```
##    <chr>        <chr>                <dbl>        <dbl> <chr>
##  1 NOR          Norway                  10       49359. Advanced Capitali~
##  2 IRL          Ireland                 10       41036. Advanced Capitali~
##  3 CHE          Switzerland             10       37581. Advanced Capitali~
##  4 NLD          Netherlands             10       36956. Advanced Capitali~
##  5 AUT          Austria                 10       35536. Advanced Capitali~
##  6 DNK          Denmark                 10       34905. Advanced Capitali~
##  7 SWE          Sweden                  10       34090. Advanced Capitali~
##  8 GBR          UK                      10       33717. Advanced Capitali~
##  9 BEL          Belgium                  8       33399. Advanced Capitali~
## 10 FIN          Finland                 10       33324. Advanced Capitali~
## # ... with 76 more rows
```

```
# 5 (top_n)
# 5. With the above data (or data of 3), find top 5 countries in terms of GDP per capita

data_country %>% top_n(5, wt = GDP_pcap_ppp)
```

```
## # A tibble: 5 x 5
##   country_code country_name polity2_score GDP_pcap_ppp country_category
##   <chr>        <chr>                <dbl>        <dbl> <chr>
## 1 AUT          Austria                 10       35536. Advanced Capitalist
## 2 CHE          Switzerland             10       37581. Advanced Capitalist
## 3 IRL          Ireland                 10       41036. Advanced Capitalist
## 4 NLD          Netherlands             10       36956. Advanced Capitalist
## 5 NOR          Norway                  10       49359. Advanced Capitalist
```

```
data_country %>% top_n(5, wt = -GDP_pcap_ppp)
```

```
## # A tibble: 5 x 5
##   country_code country_name polity2_score GDP_pcap_ppp country_category
##   <chr>        <chr>                <dbl>        <dbl> <chr>
## 1 BGD          Bangladesh              -6        1172. Asia/Mideast
## 2 MLI          Mali                     6        1023. Africa
## 3 MOZ          Mozambique               6         758. Africa
## 4 NER          Niger                    6         592. Africa
## 5 TZA          Tanzania                 1        1141. Africa
```

New: Summarize dataset

- `group_by(data, x)`: Group data into rows with the same value of `x`.
- `summarise(data, avg = mean(x))`: Summarise data into single row of values
- `summarise_all(data, funs(mean, var, sd))`: Apply summary function to each column. Example here: mean, variance, standard deviation
- `summarise_at(data, vars(x, y), funs(mean, var, sd))`: Apply summary function indicated in `funs()` to a set of variables indicated in `vars()`

```
# Getting started
data_country %>% group_by(country_category) %>%
  summarise(avg_GDP_pcap_ppp = mean(GDP_pcap_ppp))
```

```
## # A tibble: 5 x 2
##   country_category    avg_GDP_pcap_ppp
##   <chr>                          <dbl>
## 1 Advanced Capitalist           33225.
```

```
## 2 Africa                          3658.
## 3 Asia/Mideast                      NA
## 4 Latin America                   7874.
## 5 Post Communist                 12941.
```

```
# Anything special? How to handle it?
data_country %>% group_by(country_category) %>%
  summarise(avg_GDP_pcap_ppp = mean(GDP_pcap_ppp, na.rm = T))
```

```
## # A tibble: 5 x 2
##   country_category    avg_GDP_pcap_ppp
##   <chr>                          <dbl>
## 1 Advanced Capitalist           33225.
## 2 Africa                         3658.
## 3 Asia/Mideast                  10251.
## 4 Latin America                  7874.
## 5 Post Communist                12941.
```

```
# Want more than one summary statistics
data_country  %>% group_by(country_category) %>%
  summarise(GDP_pcap_ppp_avg = mean(GDP_pcap_ppp, na.rm = T),
            GDP_pcap_ppp_var = var(GDP_pcap_ppp, na.rm = T),
            GDP_pcap_ppp_sd = sd(GDP_pcap_ppp, na.rm = T),
            nrow = n())
```

```
## # A tibble: 5 x 5
##   country_category GDP_pcap_ppp_avg GDP_pcap_ppp_var GDP_pcap_ppp_sd  nrow
##   <chr>                       <dbl>            <dbl>           <dbl> <int>
## 1 Advanced Capita~           33225.        38643421.           6216.    18
## 2 Africa                      3658.        16337219.           4042.    15
## 3 Asia/Mideast               10251.        94356930.           9714.    15
## 4 Latin America               7874.        12067090.           3474.    19
## 5 Post Communist             12941.        45693673.           6760.    19
```

```
# Get multiple summary statistics conveniently with summarise_all, summarise_at
data_country %>% select(country_category, GDP_pcap_ppp) %>%
  group_by(country_category) %>%
  summarise_all(funs(mean, var, sd))
```

```
## # A tibble: 5 x 4
##   country_category      mean       var    sd
##   <chr>                <dbl>     <dbl> <dbl>
## 1 Advanced Capitalist 33225. 38643421. 6216.
## 2 Africa               3658. 16337219. 4042.
## 3 Asia/Mideast           NA        NA   NaN
## 4 Latin America        7874. 12067090. 3474.
## 5 Post Communist      12941. 45693673. 6760.
```

```
data_country %>% select(country_category, GDP_pcap_ppp, polity2_score) %>%
  group_by(country_category) %>%
  summarise_all(
    funs(mean(., na.rm = T), var(., na.rm = T), sd(., na.rm = T)))
```

```
## # A tibble: 5 x 7
##   country_category    GDP_pcap_ppp_mean polity2_score_me~ GDP_pcap_ppp_var
##   <chr>                           <dbl>             <dbl>            <dbl>
## 1 Advanced Capitalist            33225.              9.83        38643421.
```

```
## 2 Africa                          3658.              5.93        16337219.
## 3 Asia/Mideast                   10251.              4.45        94356930.
## 4 Latin America                   7874.              8.05        12067090.
## 5 Post Communist                 12941.              8.47        45693673.
## # ... with 3 more variables: polity2_score_var <dbl>,
## #   GDP_pcap_ppp_sd <dbl>, polity2_score_sd <dbl>
```

```
mean(c(1, 2, 3, 4, NA))
```

```
## [1] NA
```

```
mean(c(1, 2, 3, 4, NA), na.rm = T)
```

```
## [1] 2.5
```

```
data_country %>% group_by(country_category) %>%
  summarise_at(vars(GDP_pcap_ppp, polity2_score),
               funs(min, median, max, mean, var, sd))
```

```
## # A tibble: 5 x 13
##   country_category    GDP_pcap_ppp_min polity2_score_m~ GDP_pcap_ppp_medi~
##   <chr>                          <dbl>            <dbl>              <dbl>
## 1 Advanced Capitalist          21169.                8             33362.
## 2 Africa                         592.               -2              1456.
## 3 Asia/Mideast                    NA                -6                NA
## 4 Latin America                 2427.                5              7400.
## 5 Post Communist                2409.                5             13873.
## # ... with 9 more variables: polity2_score_median <dbl>,
## #   GDP_pcap_ppp_max <dbl>, polity2_score_max <dbl>,
## #   GDP_pcap_ppp_mean <dbl>, polity2_score_mean <dbl>,
## #   GDP_pcap_ppp_var <dbl>, polity2_score_var <dbl>,
## #   GDP_pcap_ppp_sd <dbl>, polity2_score_sd <dbl>
```

```
#
data_country %>% group_by(country_category) %>%
  mutate(avg_GDP_pcap_ppp = mean(GDP_pcap_ppp)) %>%
  select(country_code, GDP_pcap_ppp, avg_GDP_pcap_ppp, country_category)
```

```
## # A tibble: 86 x 4
## # Groups:   country_category [5]
##    country_code GDP_pcap_ppp avg_GDP_pcap_ppp country_category
##    <chr>               <dbl>            <dbl> <chr>
##  1 AGO                 5085.            3658. Africa
##  2 ALB                 6707.           12941. Post Communist
##  3 ARG                12502.            7874. Latin America
##  4 AUS                32735.           33225. Advanced Capitalist
##  5 AUT                35536.           33225. Advanced Capitalist
##  6 BEL                33399.           33225. Advanced Capitalist
##  7 BEN                 1239.            3658. Africa
##  8 BGD                 1172.              NA  Asia/Mideast
##  9 BGR                10529.           12941. Post Communist
## 10 BOL                 3972.            7874. Latin America
## # ... with 76 more rows
```

Exercise: Get minimum, maximum and median of variables GDP_pcap_ppp and polity2_score by `country_category`. Follow the codes above.

```
# Your code here
```

General advice: Keep the original, raw data; Make informative names; Handle missing values carefully.

New: Merging two datasets

An critical data management tool. Not intellectually challenging, but tedious.

The Basics

Useful functions merging datasets from `dplyr` include `left_join`, `right_join`, `full_join`, `inner_join`.

- `left_join(a, b, by = "x1")`: Join matching rows from b to a. (Keep all in `a`)
- `right_join(a, b, by = "x1")`: Join matching rows from a to b. (keep all in `b`)
- `full_join(a, b, by = "x1")`: Join data. Retain all values, all rows. (keep all – either in `a` or `b`)
- `inner_join(a, b, by = "x1")`: Join data. Retain only rows in both sets. (only keep those in both `a` and `b`)

```
# Left join
merge_left <- data_country %>% left_join(data_party, by = "country_code")
dim(merge_left)
```

```
## [1] 478  11
```

```
# Right join
merge_right <- data_country %>% right_join(data_party, by = "country_code")
dim(merge_right)
```

```
## [1] 495  11
```

```
# Full join
merge_full <- data_country %>% full_join(data_party, by = "country_code")
dim(merge_full)
```

```
## [1] 498  11
```

```
# Inner join
merge_inner <- data_country %>% inner_join(data_party, by = "country_code")
dim(merge_inner)
```

```
## [1] 475  11
```

Aside 1: Matching on multiple identifiers?

```
data_country %>% left_join(data_party, by = c("country_name", "country_code"))
```

```
## # A tibble: 478 x 10
##    country_code country_name polity2_score GDP_pcap_ppp country_category
##    <chr>        <chr>                <dbl>        <dbl> <chr>
##  1 AGO          Angola                  -2        5085. Africa
##  2 AGO          Angola                  -2        5085. Africa
##  3 AGO          Angola                  -2        5085. Africa
##  4 ALB          Albania                  9        6707. Post Communist
##  5 ALB          Albania                  9        6707. Post Communist
##  6 ALB          Albania                  9        6707. Post Communist
##  7 ALB          Albania                  9        6707. Post Communist
```

```
##  8 ALB          Albania                  9        6707. Post Communist
##  9 ARG          Argentina                8       12502. Latin America
## 10 ARG          Argentina                8       12502. Latin America
## # ... with 468 more rows, and 5 more variables: party_name <chr>,
## #   local_presence <dbl>, client_effort <dbl>, left_right <dbl>,
## #   partysize <dbl>
```

Aside 2: Matching on different variable names?

```
# Different variable names
data_country %>% left_join(data_party, by = c("country_name" = "country_name",
                                              "country_code" = "country_code"))
```

```
## # A tibble: 478 x 10
##    country_code country_name polity2_score GDP_pcap_ppp country_category
##    <chr>        <chr>                <dbl>        <dbl> <chr>
##  1 AGO          Angola                  -2        5085. Africa
##  2 AGO          Angola                  -2        5085. Africa
##  3 AGO          Angola                  -2        5085. Africa
##  4 ALB          Albania                  9        6707. Post Communist
##  5 ALB          Albania                  9        6707. Post Communist
##  6 ALB          Albania                  9        6707. Post Communist
##  7 ALB          Albania                  9        6707. Post Communist
##  8 ALB          Albania                  9        6707. Post Communist
##  9 ARG          Argentina                8       12502. Latin America
## 10 ARG          Argentina                8       12502. Latin America
## # ... with 468 more rows, and 5 more variables: party_name <chr>,
## #   local_presence <dbl>, client_effort <dbl>, left_right <dbl>,
## #   partysize <dbl>
```

Advanced: Checking what match and what do not, and debug

Checking what do not match: A command good for checking `anti_join` - `anti_join(a, b, by = "x1")`: All rows in a that do not have a match in b.

```
# What countries cannot be matched with party info?
data_country %>% anti_join(data_party, by = "country_code")
```

```
## # A tibble: 3 x 5
##   country_code country_name polity2_score GDP_pcap_ppp country_category
##   <chr>        <chr>                <dbl>        <dbl> <chr>
## 1 BRA          Brazil                   8        9034. Latin America
## 2 ROU          Romania                  9       10750. Post Communist
## 3 SRB          Serbia                   8       10128. Post Communist
```

```
data_party %>% anti_join(data_country, by = "country_code")
```

```
## # A tibble: 20 x 7
##    country_code country_name party_name local_presence client_effort
##    <chr>        <chr>        <chr>               <dbl>         <dbl>
##  1 CAN          Canada       LIB                   2.2           6
##  2 CAN          Canada       NDP                   2.33          5.38
##  3 CAN          Canada       Green                 2.83          5.38
##  4 CAN          Canada       CON                   2.2           6
##  5 CAN          Canada       BQ                    2.83          5.38
##  6 ROM          Romania      PSD                   1.21         16
```

```
##  7 ROM          Romania      PDL                  1.29         14.2
##  8 ROM          Romania      PRM                  2.23         12.4
##  9 ROM          Romania      PC                   2.43         14.7
## 10 ROM          Romania      PNL                  1.36         14.1
## 11 ROM          Romania      UDMR                 2.14         13.7
## 12 YUG          Serbia       SPS                  1.89         13.7
## 13 YUG          Serbia       DSS                  1.78         11.6
## 14 YUG          Serbia       DS                   1            14.5
## 15 YUG          Serbia       SRP                  1.2          13.3
## 16 YUG          Serbia       LDP                  2.38         10.3
## 17 YUG          Serbia       G17+                 1.57         15.4
## 18 YUG          Serbia       NS                   2.43         14.4
## 19 USA          USA          Rep                  2.13         10.5
## 20 USA          USA          Dem                  2.2           9.80
## # ... with 2 more variables: left_right <dbl>, partysize <dbl>
```

```
# Exercise: What parties cannot be matched with country info?
```

General Advice

Be extremely careful in this step. It can influence the robustness of your results in unexpected ways.

- Take good notes of what match and what do not match in the various datasets you merge together.
- Keep all raw data
- Keep a full dataset, which is a `full_join` of all.
- Keep a "smallest" dataset, usually the one you use to fit your full model