DataFest R Workshop May 2020

Nathalie Moon (adapted from Sotirios Damouras' 2019 Workshop)

May 17, 2020

PART 1

Task 1.1

Create an R project & associated folder for the workshop

Task 1.2

Download and install the 'tidyverse' library

Task 1.3

Download the "dinesafe.csv" file from the workshop folder and save it in a project subfolder called "data". Then read the data into a data-frame called "dinesafe"

```
dinesafe <- read_csv("./data/dinesafe.csv")
```

```
## Parsed with column specification:
## cols(
##   ROW_ID = col_double(),
##   ESTABLISHMENT_ID = col_double(),
##   INSPECTION_ID = col_double(),
##   ESTABLISHMENT_NAME = col_character(),
##   ESTABLISHMENTTYPE = col_character(),
##   ESTABLISHMENT_ADDRESS = col_character(),
##   LATITUDE = col_double(),
##   LONGITUDE = col_double(),
##   ESTABLISHMENT_STATUS = col_character(),
##   MINIMUM_INSPECTIONS_PERYEAR = col_double(),
##   INFRACTION_DETAILS = col_character(),
##   INSPECTION_DATE = col_date(format = ""),
##   SEVERITY = col_character(),
##   ACTION = col_character(),
##   COURT_OUTCOME = col_character(),
##   AMOUNT_FINED = col_double()
## )
```

Task 1.4

Use the `glimpse()` function to take a look at the `dinesafe` data frame's structure

```
glimpse(dinesafe)
```

```
## Rows: 90,520
```

```
## Columns: 16
## $ ROW_ID                       <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, ...
## $ ESTABLISHMENT_ID             <dbl> 1222579, 1222579, 1222579, 1222579, 122...
## $ INSPECTION_ID                <dbl> 103868579, 104063869, 104246429, 104246...
## $ ESTABLISHMENT_NAME           <chr> "SAI-LILA KHAMAN DHOKLA HOUSE", "SAI-LI...
## $ ESTABLISHMENTTYPE            <chr> "Food Take Out", "Food Take Out", "Food...
## $ ESTABLISHMENT_ADDRESS        <chr> "870 MARKHAM RD", "870 MARKHAM RD", "87...
## $ LATITUDE                     <dbl> 43.76798, 43.76798, 43.76798, 43.76798,...
## $ LONGITUDE                    <dbl> -79.22903, -79.22903, -79.22903, -79.22...
## $ ESTABLISHMENT_STATUS         <chr> "Pass", "Pass", "Pass", "Pass", "Pass",...
## $ MINIMUM_INSPECTIONS_PERYEAR <dbl> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, ...
## $ INFRACTION_DETAILS           <chr> "Operator fail to properly wash equipme...
## $ INSPECTION_DATE              <date> 2016-12-21, 2017-10-04, 2018-06-20, 20...
## $ SEVERITY                     <chr> "M - Minor", NA, "NA - Not Applicable",...
## $ ACTION                       <chr> "Notice to Comply", NA, "Notice to Comp...
## $ COURT_OUTCOME                <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
## $ AMOUNT_FINED                 <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
```

Task 1.5

Use the `View()` function to view the `dinesafe` data frame as a spreadsheet

```
View(dinesafe)
```

Part 2

Task 2.1

Find all distinct establishment types. Hints: Are values of establishment in different rows or different columns? Which of the `dplyr` functions can you use to remove duplicated values?

```
dinesafe %>%
  distinct(ESTABLISHMENTTYPE)
```

```
## # A tibble: 55 x 1
##    ESTABLISHMENTTYPE
##    <chr>
##  1 Food Take Out
##  2 Restaurant
##  3 Cafeteria
##  4 Commissary
##  5 Private Club
##  6 Child Care - Catered
##  7 Food Store (Convenience / Variety)
##  8 Child Care - Food Preparation
##  9 Food Depot
## 10 Food Court Vendor
## # ... with 45 more rows
```

Task 2.2

Find all inspections that took place on August 21st, 2018 (i.e. "2018-08-21"). Hints: (1) What variable contains the date of inspections? (2) Which `dplyr` function can you use to keep only observations for establishments that got inspected on this date?

```
dinesafe %>%
  filter(INSPECTION_DATE == "2018-08-21") %>%
  distinct(ESTABLISHMENT_ID, .keep_all = TRUE) %>%   ### What happens if you exclude this line?
  select(INSPECTION_DATE, ESTABLISHMENT_NAME)
```

```
## # A tibble: 100 x 2
##    INSPECTION_DATE ESTABLISHMENT_NAME
##    <date>          <chr>
##  1 2018-08-21      BARDI'S STEAK HOUSE TAVERN
##  2 2018-08-21      BRAZIL BAKERY & PASTRY
##  3 2018-08-21      LEASIDE RETIREMENT RESIDENCE
##  4 2018-08-21      ISLAND FOODS
##  5 2018-08-21      IZBA RESTAURANT
##  6 2018-08-21      THE KEG MANSION RESTAURANT
##  7 2018-08-21      MEZZA NOTTE TRATTORIA
##  8 2018-08-21      MILLWOOD JUNIOR Y
##  9 2018-08-21      MON SHEONG HOME FOR THE AGED
## 10 2018-08-21      SILVER FOUNTAIN FAST FOOD
## # ... with 90 more rows
```

Task 2.3

Find the total # of distinct inspections

```
dinesafe %>%
  summarise(n_distinct(INSPECTION_ID))
```

```
## # A tibble: 1 x 1
##   `n_distinct(INSPECTION_ID)`
##                         <int>
## 1                       55589
```

```
# or

dinesafe %>%
  distinct(INSPECTION_ID) %>%
  summarise(n())
```

```
## # A tibble: 1 x 1
##   `n()`
##   <int>
## 1 55589
```

Task 2.4

Rank establishment types by total amount fined

```
dinesafe %>%
  group_by(ESTABLISHMENTTYPE) %>%
  summarise(TOTAL_AMOUNT = sum(AMOUNT_FINED, na.rm = TRUE)) %>%
  arrange(desc(TOTAL_AMOUNT))
```

```
## # A tibble: 55 x 2
##    ESTABLISHMENTTYPE                   TOTAL_AMOUNT
##    <chr>                                      <dbl>
##  1 Restaurant                                40952
##  2 Food Take Out                              6360.
##  3 Food Court Vendor                          3460
##  4 Supermarket                                2970
##  5 Food Store (Convenience / Variety)         1540
##  6 Bakery                                     1225
##  7 Food Depot                                  465
##  8 Food Processing Plant                       240
##  9 Butcher Shop                                235
## 10 Cafeteria                                    60
## # ... with 45 more rows
```

Task 2.5

Rank establishment types by average amount fined per establishment. Hints: (1) What variable might you want to group on? (2) How can you calculate the average amount fined in each category?

```
dinesafe %>%
  group_by(ESTABLISHMENTTYPE) %>%
  summarise(TOTAL_AMOUNT = sum(AMOUNT_FINED, na.rm = TRUE),
            N_EST = n_distinct(ESTABLISHMENT_ID)) %>%
  mutate( AVG_AMOUNT = TOTAL_AMOUNT / N_EST ) %>%
  arrange(desc(AVG_AMOUNT))
```

```
## # A tibble: 55 x 4
##    ESTABLISHMENTTYPE                   TOTAL_AMOUNT N_EST AVG_AMOUNT
##    <chr>                                      <dbl> <int>      <dbl>
##  1 Food Court Vendor                          3460    467      7.41
##  2 Supermarket                                2970    439      6.77
```

```
##  3 Restaurant                              40952   6977      5.87
##  4 Bakery                                   1225    381      3.22
##  5 Food Depot                                465    171      2.72
##  6 Food Take Out                            6360.  2524      2.52
##  7 Butcher Shop                              235    157      1.50
##  8 Food Processing Plant                     240    207      1.16
##  9 Food Store (Convenience / Variety)       1540   2034      0.757
## 10 Cafeteria                                 60    186      0.323
## # ... with 45 more rows
```

Task 2.6 (Challenging)

Find the establishment with the highest non-zero total fine amount whithin each establishment type. Hints: (1) Start by calculating the total fine for each establishment (think of what variable to group on to achieve this), then create new groups based on `ESTABLISHMENTTYPE` and use one of the `dplyr` functions to keep only the observations with the highest total in each of these new groups. Note - you'll need to use `ungroup` before regrouping the data into new groups.

```
dinesafe %>%
  group_by(ESTABLISHMENTTYPE, ESTABLISHMENT_NAME, ESTABLISHMENT_ID) %>%
  summarise(TOTAL_AMOUNT = sum(AMOUNT_FINED, na.rm = TRUE)) %>%
  filter(TOTAL_AMOUNT > 0) %>%
  ungroup() %>%
  group_by(ESTABLISHMENTTYPE) %>%
  top_n(1, TOTAL_AMOUNT)
```

```
## # A tibble: 10 x 4
## # Groups:   ESTABLISHMENTTYPE [10]
##    ESTABLISHMENTTYPE        ESTABLISHMENT_NAME     ESTABLISHMENT_ID TOTAL_AMOUNT
##    <chr>                    <chr>                             <dbl>        <dbl>
##  1 Bakery                   GLOUCESTER BAKERY              10512360          700
##  2 Butcher Shop             AL-FATEH GROCERS & HA~          9394641          120
##  3 Cafeteria                ROGERS CAFETERIA               10542102           60
##  4 Food Court Vendor        FRESH & DELICIOUS FAS~         10387207         2000
##  5 Food Depot               NUTRIFRESH JUICE DIST~          9419024          465
##  6 Food Processing Plant    Shang Hai Frozen Food          10532113          240
##  7 Food Store (Convenience~ AUTHENTIC FOOD PLUS M~         10578885          875
##  8 Food Take Out            BERNARD'S PILIPINO SP~          9001425         1320.
##  9 Restaurant               OLD SCHOOL                     10531573         3625
## 10 Supermarket              FOOD DEPOT SUPERMARKET          9047834         1300
```

PART 3

The file data/establishments.csv contains information on different establishments, in particular its neighborhood.

```
# Read this file in R
establishments <- read_csv("data/establishments.csv")
```

```
## Parsed with column specification:
## cols(
##   ESTABLISHMENT_ID = col_double(),
##   ESTABLISHMENT_NAME = col_character(),
##   ESTABLISHMENT_ADDRESS = col_character(),
##   ESTABLISHMENT_NEIGHBORHOOD = col_character()
## )
```

```
glimpse(establishments)
```

```
## Rows: 15,476
## Columns: 4
## $ ESTABLISHMENT_ID           <dbl> 10403532, 10571762, 10387622, 10635390, ...
## $ ESTABLISHMENT_NAME         <chr> "JAY EXCLUSIVE CATERERS", "THE CAPTAIN'S...
## $ ESTABLISHMENT_ADDRESS      <chr> "1129 BROADVIEW AVE", "671 COLLEGE ST", ...
## $ ESTABLISHMENT_NEIGHBORHOOD <chr> "Brookhaven-Amesbury", "Lansing-Westgate...
```

We will try to match this information with the dinesafe data.

Note that NOT ALL inspected establishments are present.

Task 3.1

Do an inner_join between the dinesafe and establishments tables. Hint: Which variable will you use to do the matching (it should be a variable which is present in both datasets)

```
dinesafe %>%
  inner_join( establishments, by = "ESTABLISHMENT_ID" )
```

```
## # A tibble: 85,643 x 19
##    ROW_ID ESTABLISHMENT_ID INSPECTION_ID ESTABLISHMENT_N~ ESTABLISHMENTTY~
##     <dbl>            <dbl>         <dbl> <chr>            <chr>
##  1      1          1222579     103868579 SAI-LILA KHAMAN~ Food Take Out
##  2      2          1222579     104063869 SAI-LILA KHAMAN~ Food Take Out
##  3      3          1222579     104246429 SAI-LILA KHAMAN~ Food Take Out
##  4      4          1222579     104246429 SAI-LILA KHAMAN~ Food Take Out
##  5      5          1222579     104246429 SAI-LILA KHAMAN~ Food Take Out
##  6      6          1222579     104277664 SAI-LILA KHAMAN~ Food Take Out
##  7      7          1222579     104277664 SAI-LILA KHAMAN~ Food Take Out
##  8      8          1222807     103874297 PHO BO TO        Restaurant
##  9      9          1222807     103941166 PHO BO TO        Restaurant
## 10     10          1222807     104018926 PHO BO TO        Restaurant
## # ... with 85,633 more rows, and 14 more variables:
## #   ESTABLISHMENT_ADDRESS.x <chr>, LATITUDE <dbl>, LONGITUDE <dbl>,
## #   ESTABLISHMENT_STATUS <chr>, MINIMUM_INSPECTIONS_PERYEAR <dbl>,
## #   INFRACTION_DETAILS <chr>, INSPECTION_DATE <date>, SEVERITY <chr>,
## #   ACTION <chr>, COURT_OUTCOME <chr>, AMOUNT_FINED <dbl>,
## #   ESTABLISHMENT_NAME.y <chr>, ESTABLISHMENT_ADDRESS.y <chr>,
```

```
## #   ESTABLISHMENT_NEIGHBORHOOD <chr>
```

Task 3.2

Use inner_join to rank the neighborhoods by the number of "C - Crucial" type infractions in restaurants. Hint: Either before or after joining, you'll need to use the `filter` function to keep only the observations we're interested in here (e.g. restaurants with inspection results of "C - Crucial"). After joining, think about which variable to group by before sorting.

```
dinesafe %>%
  filter( ESTABLISHMENTTYPE == "Restaurant",
          SEVERITY == "C - Crucial") %>%
  inner_join(establishments, by = "ESTABLISHMENT_ID" )  %>%
  group_by(ESTABLISHMENT_NEIGHBORHOOD) %>%
  summarise( N_CRUCIAL_INFR = n() ) %>%
  arrange( desc(N_CRUCIAL_INFR) )
```

```
## # A tibble: 108 x 2
##    ESTABLISHMENT_NEIGHBORHOOD        N_CRUCIAL_INFR
##    <chr>                                      <int>
##  1 Rexdale-Kipling                               76
##  2 St.Andrew-Windfields                          74
##  3 Bayview Village                               73
##  4 Pelmo Park-Humberlea                          53
##  5 Wychwood                                      51
##  6 West Hill                                     50
##  7 Casa Loma                                     47
##  8 Mount Olive-Silverstone-Jamestown             40
##  9 South Parkdale                                37
## 10 East End-Danforth                             31
## # ... with 98 more rows
```

Task 3.3

Find which (distinct) establishments did NOT get matched to a neighborhood. Hint: Use `anti_join`

```
dinesafe %>%
  distinct( ESTABLISHMENT_ID, .keep_all = TRUE ) %>%
  anti_join( establishments, by = "ESTABLISHMENT_ID" )
```

```
## # A tibble: 815 x 16
##    ROW_ID ESTABLISHMENT_ID INSPECTION_ID ESTABLISHMENT_N~ ESTABLISHMENTTY~
##     <dbl>            <dbl>         <dbl> <chr>            <chr>
##  1     17          9000004     103927199 PAPINO'S PIZZA   Food Take Out
##  2     20          9000026     103930674 2-4-1 PIZZA      Food Take Out
##  3     56          9000097     103757750 30 UP CLUB       Private Club
##  4    193          9000325     103849256 RADISSON ADMIRA~ Banquet Facility
##  5    245          9000375     103858374 A.C.C. STAND #4~ Food Take Out
##  6    302          9000458     103828696 ALEX REI DOS LE~ Food Take Out
##  7    373          9000646     103864670 MCLOUGHLIN SCHO~ Child Care - Ca~
##  8    826          9001395     103823250 BENDALE ACRES K~ Institutional F~
##  9    998          9001638     103977615 BLOOR FRUIT MAR~ Food Store (Con~
## 10   1150          9001846     103895191 BRAZILIAN CANAD~ Food Processing~
## # ... with 805 more rows, and 11 more variables: ESTABLISHMENT_ADDRESS <chr>,
## #   LATITUDE <dbl>, LONGITUDE <dbl>, ESTABLISHMENT_STATUS <chr>,
## #   MINIMUM_INSPECTIONS_PERYEAR <dbl>, INFRACTION_DETAILS <chr>,
```

```
## #   INSPECTION_DATE <date>, SEVERITY <chr>, ACTION <chr>, COURT_OUTCOME <chr>,
## #   AMOUNT_FINED <dbl>
```

PART 4

Our goal is to create a publication-quality graph with ggplot2. You will try to reproduce the Gapminder World Poster on World Health: https://www.gapminder.org/downloads/updated-gapminder-world-poster-2015/ using data in the gapminder package

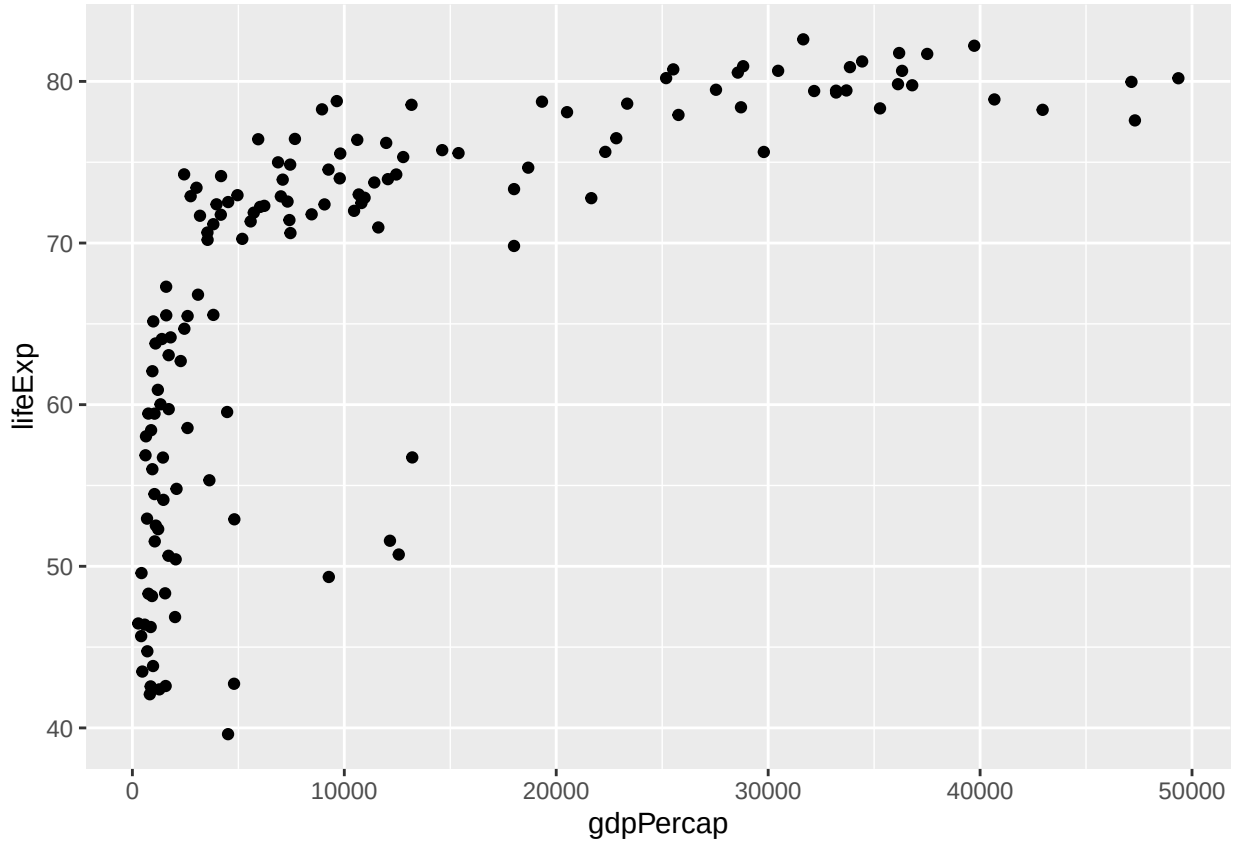
```
# load data
#install.packages("gapminder")
library(gapminder)

# Take a look at the data
glimpse(gapminder)
```

```
## Rows: 1,704
## Columns: 6
## $ country   <fct> Afghanistan, Afghanistan, Afghanistan, Afghanistan, Afgha...
## $ continent <fct> Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asi...
## $ year      <int> 1952, 1957, 1962, 1967, 1972, 1977, 1982, 1987, 1992, 199...
## $ lifeExp   <dbl> 28.801, 30.332, 31.997, 34.020, 36.088, 38.438, 39.854, 4...
## $ pop       <int> 8425333, 9240934, 10267083, 11537966, 13079460, 14880372,...
## $ gdpPercap <dbl> 779.4453, 820.8530, 853.1007, 836.1971, 739.9811, 786.113...
```

Task 4.1

Create a scatter-plot of Life Expectancy (lifeExp) versus GDP-per-capita (gdpPercap), for 2007 data

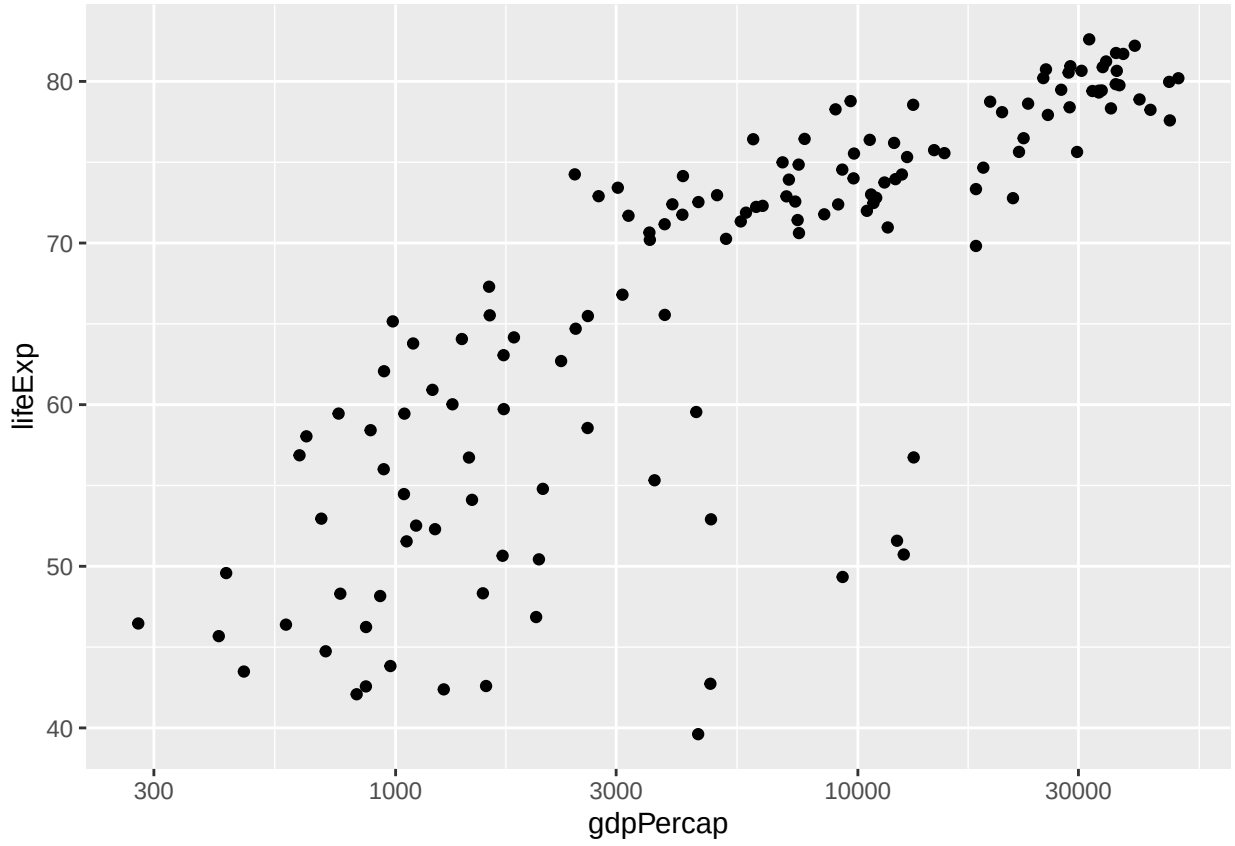
```
gapminder %>%
  filter(year == 2007) %>%
  ggplot(aes(y = lifeExp, x = gdpPercap)) +
  geom_point()
```



Task 4.2

On your previous plot, change the x-axis (gdg/cap) to log-scale. Hint: look at the choices for "Scale" geometries on the `ggplot2` cheat sheet

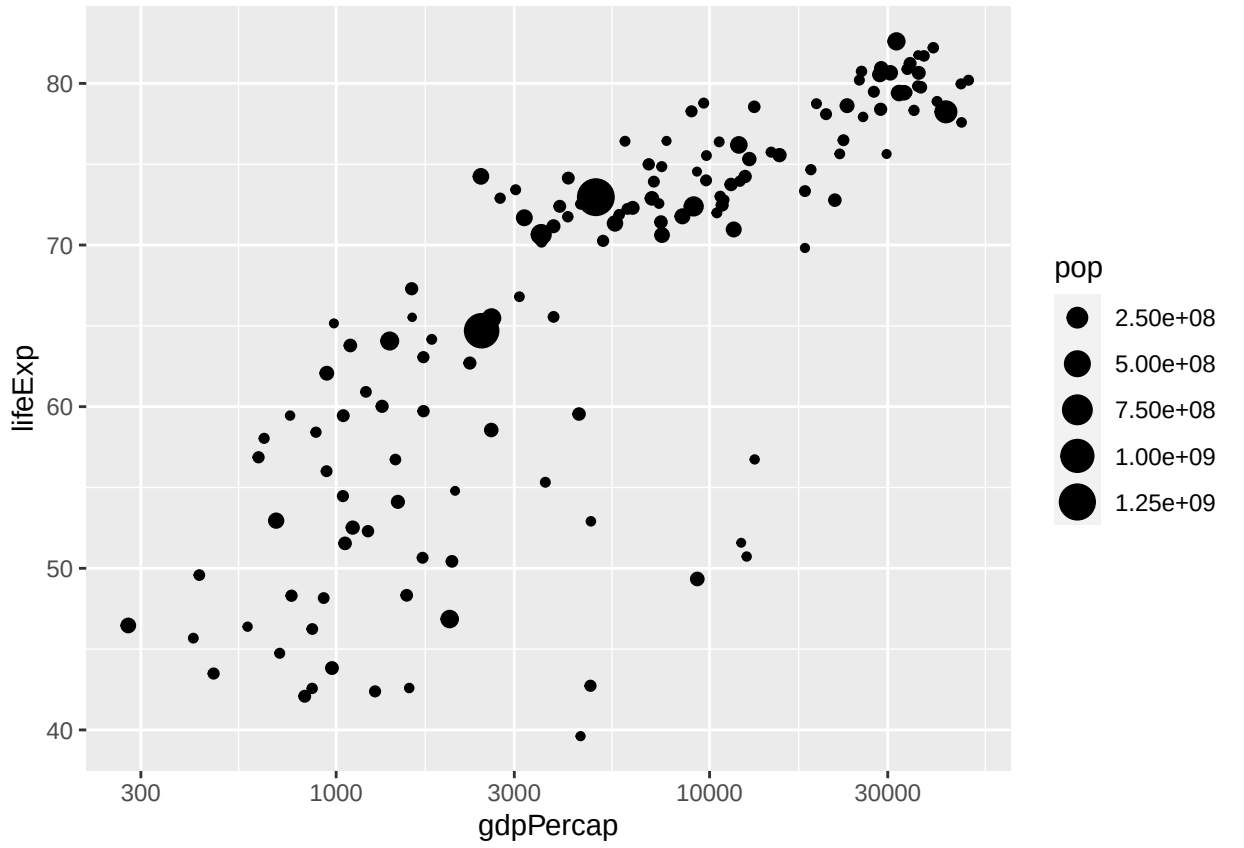
```
gapminder %>%
  filter(year == 2007) %>%
  ggplot(aes(y = lifeExp, x = gdpPercap)) +
  geom_point() +
  scale_x_log10()
```



Task 4.3

On your previous plot, change change the size of the points according to the population of each country

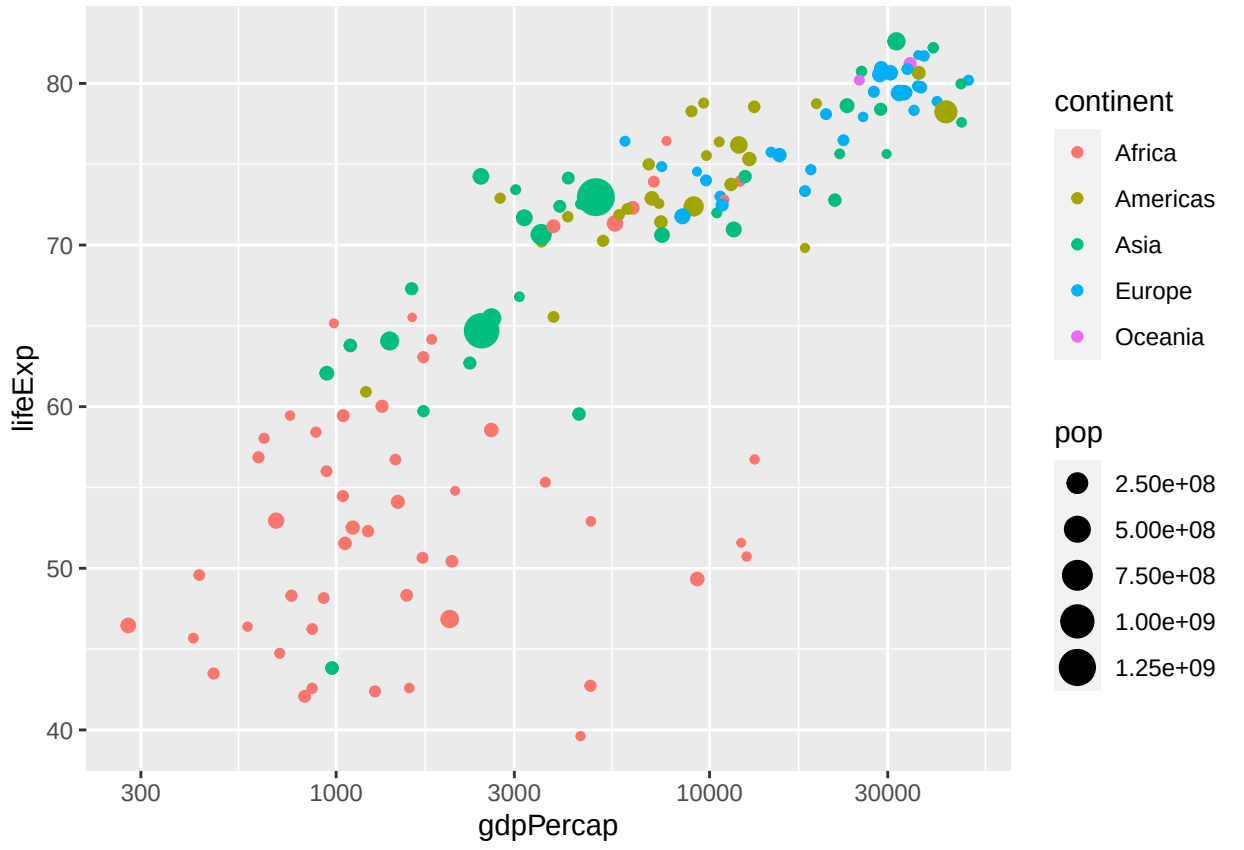
```
gapminder %>%
  filter( year == 2007) %>%
  ggplot(aes(x = gdpPercap, y=lifeExp, size = pop)) +
  geom_point() +
  scale_x_log10()
```



Task 4.4

On your previous plot, change the color of the points according to the continent

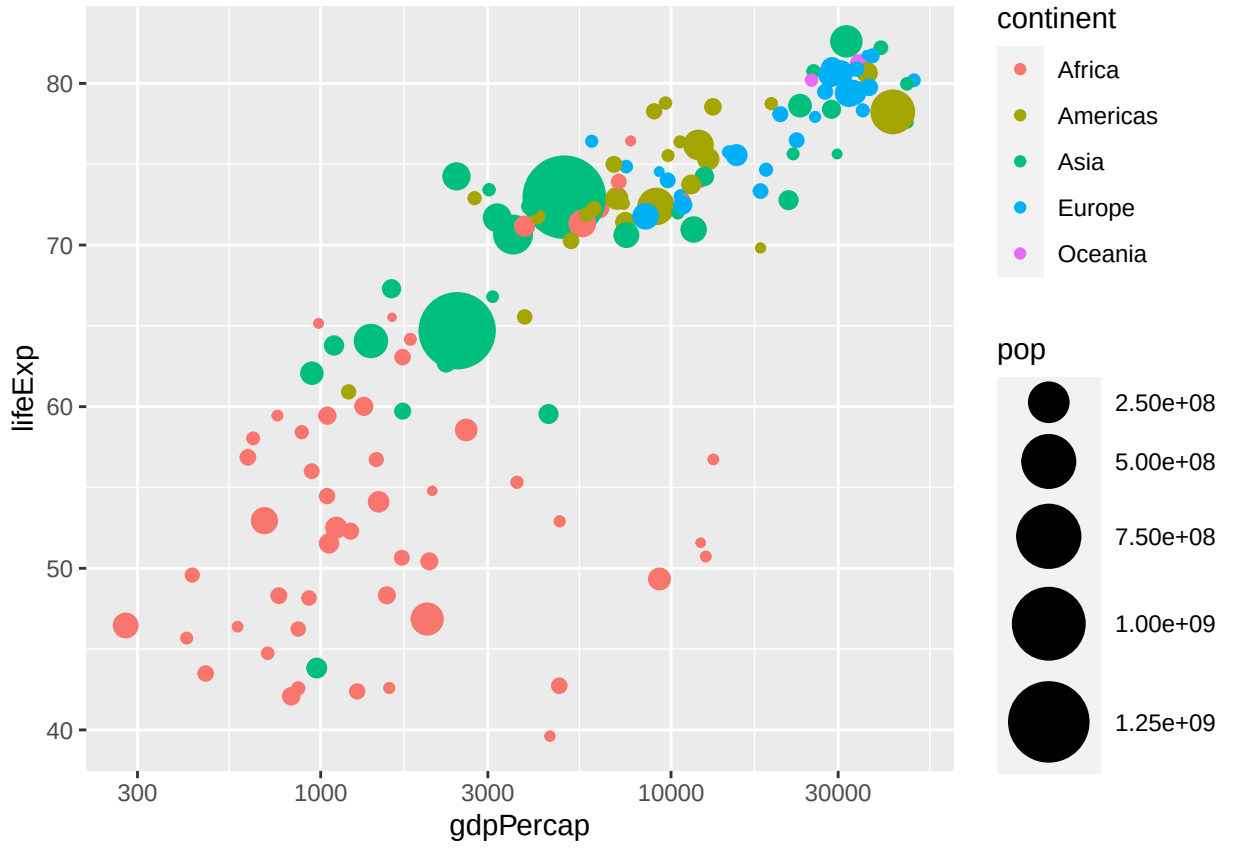
```
gapminder %>%
  filter( year == 2007) %>%
  ggplot(aes(x = gdpPercap, y=lifeExp,
             size = pop, color = continent)) +
  geom_point() +
  scale_x_log10()
```



Task 4.5

On your previous plot, change the scale of each point to range from 1 to 14. Hint: Use the `scale_size(range = )` geometry to specify the range of sizes

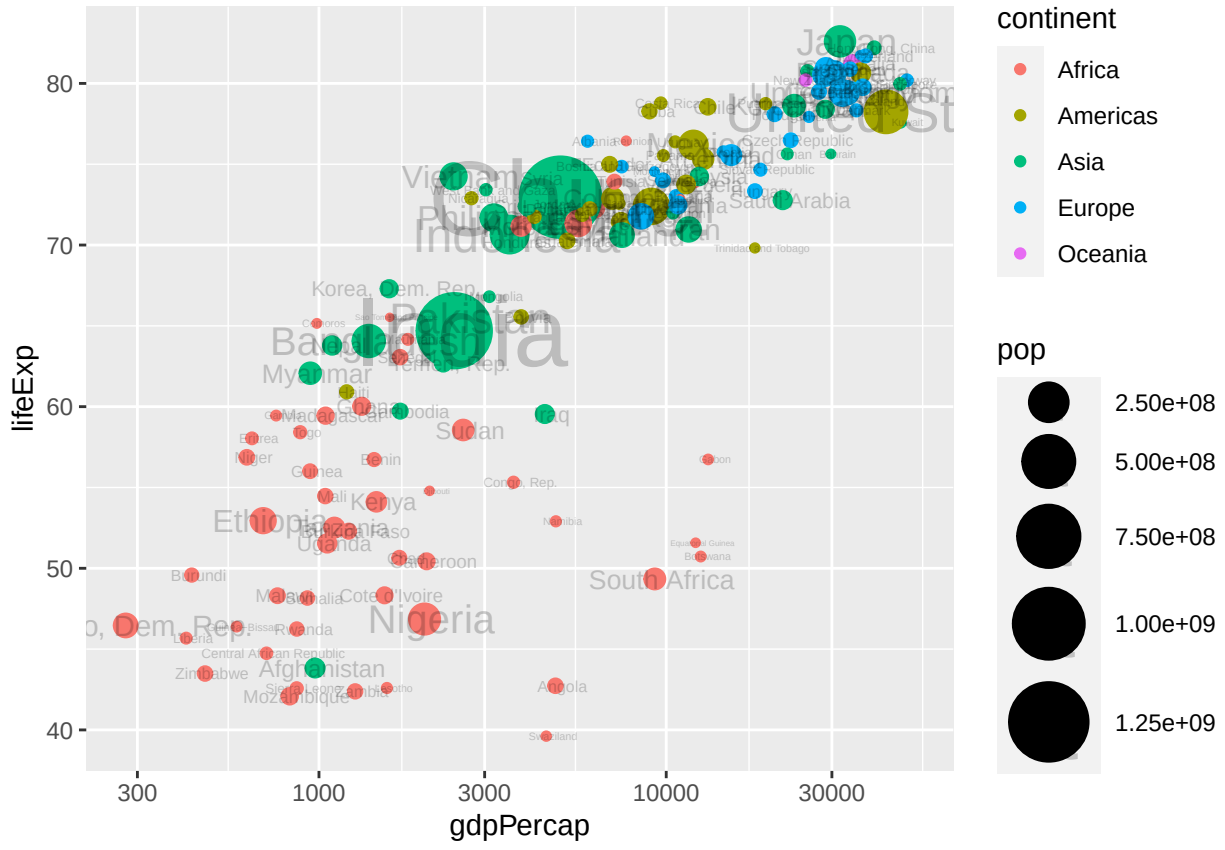
```
gapminder %>%
  filter( year == 2007) %>%
  ggplot(aes(x = gdpPercap, y=lifeExp,
             size = pop, color = continent)) +
  geom_point() +
  scale_x_log10() +
  scale_size(range = c(1,14))
```



Task 4.6

On your previous plot, label to each point with the name of the country; use options nudge_x = .02, alpha = .2 to make the labels readable.

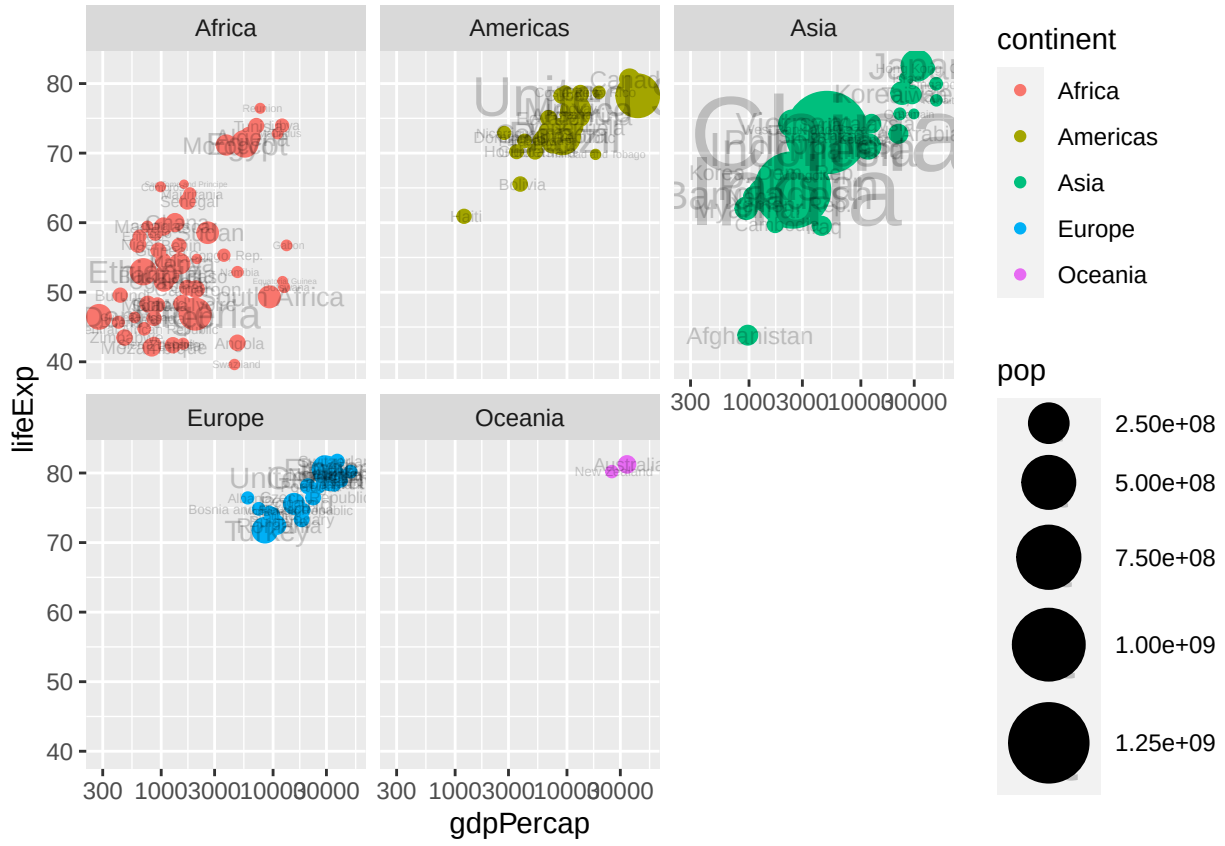
```
gapminder %>%
  filter( year == 2007) %>%
  ggplot(aes(x = gdpPercap, y=lifeExp,
             size = pop)) +
  geom_point(aes(color = continent)) +
  scale_x_log10() +
  scale_size(range = c(1,14)) +
  geom_text(aes(label = country),
            nudge_x = 0.02, alpha = .2)
```



Task 4.7

Modify the previous plot to make facets for each continent

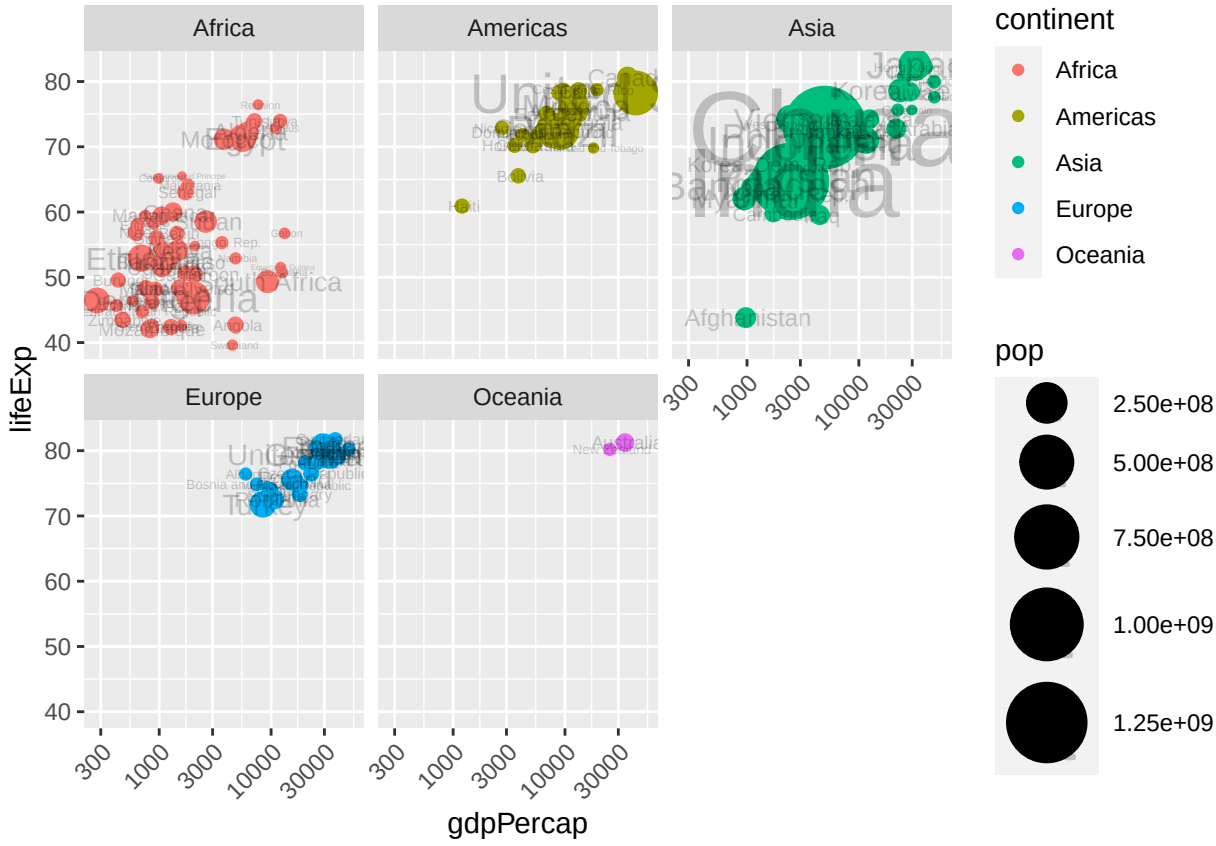
```
gapminder %>%
  filter( year == 2007) %>%
  ggplot(aes(x = gdpPercap, y=lifeExp,
             size = pop)) +
  geom_point(aes(color = continent)) +
  scale_x_log10() +
  scale_size(range = c(1,14)) +
  geom_text(aes(label = country),
            nudge_x = 0.02, alpha = .2) +
  facet_wrap(facets = ~continent)
```



Task 4.8

If your x-axis labels are hard to read due to overlapping, you may want to rotate theme. You can use the `theme(axis.text.x=element_text(angle = 90, hjust = 0))` geometry to adjust this; change the values of `angle` and `hjust` to figure out the effect of these two arguments and find a combination that makes the labels easier to read.

```
gapminder %>%
  filter( year == 2007) %>%
  ggplot(aes(x = gdpPercap, y=lifeExp,
             size = pop)) +
  geom_point(aes(color = continent)) +
  scale_x_log10() +
  scale_size(range = c(1,14)) +
  geom_text(aes(label = country),
            nudge_x = 0.02, alpha = .2) +
  facet_wrap(facets = ~continent) +
  theme(axis.text.x = element_text(angle=45, hjust=1))
```


Africa
Americas
Asia
Europe
Oceania
lifeExp
gdpPercap
80
70
60
50
40
300
1000
3000
10000
30000
continent
Africa
Americas
Asia
Europe
Oceania
pop
2.50e+08
5.00e+08
7.50e+08
1.00e+09
1.25e+09