Week 4 Exercises

Ray Chandonnet

11/20/2022

Please complete all exercises below. You may use any library that we have covered in class. The data we will be using comes from the tidyr package, so you must use that.

1) Examine the who and population data sets that come with the tidyr library. the who data is not tidy, you will need to reshape the new_sp_m014 to newrel_f65 columns to long format retaining country, iso2, iso3, and year. The data in the columns you are reshaping contains patterns described in the details section below. You will need to assign three columns: diagnosis, gender, and age to the patterns described in the details.

Your tidy data should look like the following: country iso2 iso3 year diagnosis gender age count 1 Afghanistan AF AFG 1980 sp m 014 NA 2 Afghanistan AF AFG 1980 sp m 1524 NA 3 Afghanistan AF AFG 1980 sp m 2534 NA 4 Afghanistan AF AFG 1980 sp m 3544 NA 5 Afghanistan AF AFG 1980 sp m 4554 NA 6 Afghanistan AF AFG 1980 sp m 5564 NA

Details The data uses the original codes given by the World Health Organization. The column names for columns five through 60 are made by combining new_ to a code for method of diagnosis (rel = relapse, sn = negative pulmonary smear, sp = positive pulmonary smear, ep = extrapulmonary) to a code for gender (f = female, m = male) to a code for age group (014 = 0-14 yrs of age, 1524 = 15-24 years of age, 2534 = 25 to 34 years of age, 3544 = 35 to 44 years of age, 4554 = 45 to 54 years of age, 5564 = 55 to 64 years of age, 65 = 65 years of age or older).

Note: use data(who) and data(population) to load the data into your environment. Use the arguments cols, names_to, names_pattern, and values_to. Your regex should be = ((“new_?(.)_(.)(.)“)

https://tidyr.tidyverse.org/reference/who.html

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(ggplot2)
data(who)  # load data from tidyverse
data(population) # load data from tidyverse
who <- who %>%   #use pipe to run transformation to long format
  pivot_longer(!c(country, iso2, iso3, year),
               names_to = c("diagnosis", "gender","age"),
               names_pattern = "new_?(.*)_(.)(.*)",
```

```
               values_to = "count",)
head(who)
```

```
## # A tibble: 6 x 8
##   country     iso2  iso3   year diagnosis gender age   count
##   <chr>       <chr> <chr> <int> <chr>     <chr>  <chr> <int>
## 1 Afghanistan AF    AFG    1980 sp        m      014      NA
## 2 Afghanistan AF    AFG    1980 sp        m      1524     NA
## 3 Afghanistan AF    AFG    1980 sp        m      2534     NA
## 4 Afghanistan AF    AFG    1980 sp        m      3544     NA
## 5 Afghanistan AF    AFG    1980 sp        m      4554     NA
## 6 Afghanistan AF    AFG    1980 sp        m      5564     NA
```

2) There are two common keys between the data sets, with who as the left table, join the population data by country and year so that the population is available within the who dataset.

```
who <- who %>%
  left_join(population,by=c('country','year'))
head(who)
```

```
## # A tibble: 6 x 9
##   country     iso2  iso3   year diagnosis gender age   count population
##   <chr>       <chr> <chr> <int> <chr>     <chr>  <chr> <int>      <int>
## 1 Afghanistan AF    AFG    1980 sp        m      014      NA         NA
## 2 Afghanistan AF    AFG    1980 sp        m      1524     NA         NA
## 3 Afghanistan AF    AFG    1980 sp        m      2534     NA         NA
## 4 Afghanistan AF    AFG    1980 sp        m      3544     NA         NA
## 5 Afghanistan AF    AFG    1980 sp        m      4554     NA         NA
## 6 Afghanistan AF    AFG    1980 sp        m      5564     NA         NA
```

```
who[123456,] # show a random row to demonstrate that population was pulled in
```

```
## # A tibble: 1 x 9
##   country iso2  iso3   year diagnosis gender age   count population
##   <chr>   <chr> <chr> <int> <chr>     <chr>  <chr> <int>      <int>
## 1 Eritrea ER    ERI    2000 ep        m      3544     NA    3939348
```

3) Split the age column into two columns, min age and max age. Notice that there is no character separator. Check the documentation with ?separate to understand other ways to separate the age column. Keep in mind that 0 to 14 is coded as 014 (3 characters) and the other age groups are coded with 4 characters. 65 only has two characters, but we will ignore that until the next prolem.

```
who <- separate(who, col=age,into=c("min_age","max_age"),sep=-2,remove=TRUE)
who[1:8,] #show first eight rows to prove that it worked
```

```
## # A tibble: 8 x 10
##   country     iso2  iso3   year diagnosis gender min_age max_age count populat~1
##   <chr>       <chr> <chr> <int> <chr>     <chr>  <chr>   <chr>   <int>     <int>
## 1 Afghanistan AF    AFG    1980 sp        m      "0"     14         NA        NA
## 2 Afghanistan AF    AFG    1980 sp        m      "15"    24         NA        NA
## 3 Afghanistan AF    AFG    1980 sp        m      "25"    34         NA        NA
## 4 Afghanistan AF    AFG    1980 sp        m      "35"    44         NA        NA
## 5 Afghanistan AF    AFG    1980 sp        m      "45"    54         NA        NA
## 6 Afghanistan AF    AFG    1980 sp        m      "55"    64         NA        NA
## 7 Afghanistan AF    AFG    1980 sp        m      ""      65         NA        NA
## 8 Afghanistan AF    AFG    1980 sp        f      "0"     14         NA        NA
## # ... with abbreviated variable name 1: population
```

4) Since we ignored the 65+ group in the previous problem we will fix it here. If you examine the data you will notice that 65 was placed into the max_age column and there is no value for min_age for those records. To fix this use mutate() in order to replace the blank value in the min_age column with the value from the max_age column and another mutate to replace the 65 in the max column with an Inf. Be sure to keep the variables as character vectors.

```
# Here I use the ifelse function to only change the min and max age when the max age is 65
who <- who %>%
  mutate(min_age = ifelse(max_age == "65","65",min_age),
         max_age = ifelse(max_age == "65", Inf, max_age)
  )
who[1:8,] #show first eight rows to prove that it worked
```

```
## # A tibble: 8 x 10
##   country     iso2  iso3   year diagnosis gender min_age max_age count populat~1
##   <chr>       <chr> <chr> <int> <chr>     <chr>  <chr>   <chr>   <int>     <int>
## 1 Afghanistan AF    AFG    1980 sp        m      0       14         NA        NA
## 2 Afghanistan AF    AFG    1980 sp        m      15      24         NA        NA
## 3 Afghanistan AF    AFG    1980 sp        m      25      34         NA        NA
## 4 Afghanistan AF    AFG    1980 sp        m      35      44         NA        NA
## 5 Afghanistan AF    AFG    1980 sp        m      45      54         NA        NA
## 6 Afghanistan AF    AFG    1980 sp        m      55      64         NA        NA
## 7 Afghanistan AF    AFG    1980 sp        m      65      Inf        NA        NA
## 8 Afghanistan AF    AFG    1980 sp        f      0       14         NA        NA
## # ... with abbreviated variable name 1: population
```

5) Find the count per diagnosis for males and females.

See ?sum for a hint on resolving NA values.

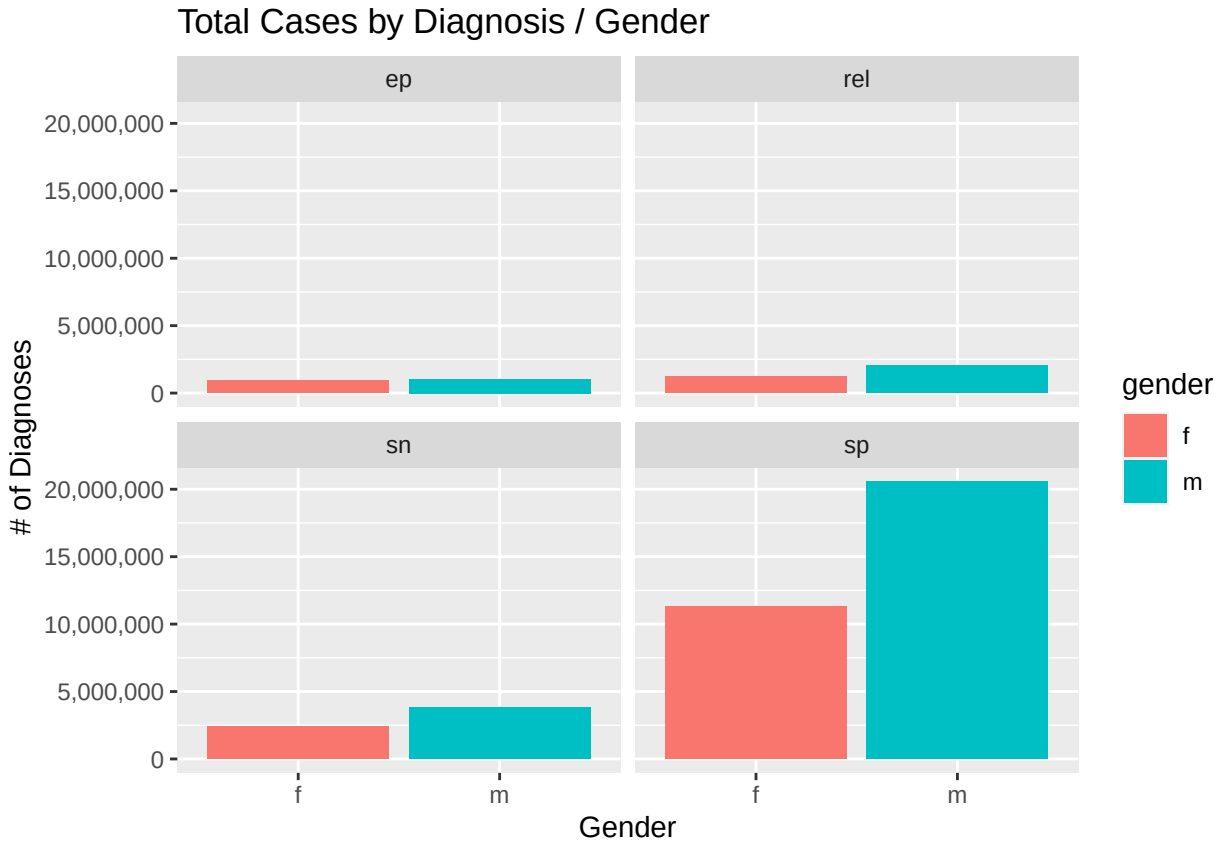
```
diagnoses_by_gender <- aggregate(who$count,
                                 by=list(who$gender, who$diagnosis),
                                 FUN = sum,
                                 na.rm=TRUE)
colnames(diagnoses_by_gender) <- c("gender", "diagnosis","count")
diagnoses_by_gender
```

```
##   gender diagnosis    count
## 1      f        ep   941880
## 2      m        ep  1044299
## 3      f       rel  1201596
## 4      m       rel  2018976
## 5      f        sn  2439139
## 6      m        sn  3840388
## 7      f        sp 11324409
## 8      m        sp 20586831
```

6) Now create a plot using ggplot and geom_col where your x axis is gender, your y axis represents the counts, and facet by diagnosis. Be sure to give your plot a title and resolve the axis labels.

```
diagnoses_by_gender %>%
  ggplot(aes(x=gender,
             y = count,
             fill=gender)) +
  geom_col(show.legend = T) +
  facet_wrap(~diagnosis) +
  xlab("Gender") +
```

```
  ylab("# of Diagnoses") +
  ggtitle("Total Cases by Diagnosis / Gender") +
  scale_y_continuous(labels = scales::comma)
```



7) Find the percentage of population by year, gender, and diagnosis. Be sure to remove rows containing NA values.

```
# I am assuming that the instructions here mean the percent of total global population
# First, create a dataframe that calculates total global population by year
global_population <- aggregate(population$population,
                               by=list(population$year),
                               FUN=sum,
                               na.rm=TRUE)
colnames(global_population) <- c("year","population")
head(global_population)
```

```
##   year population
## 1 1995 5717507165
## 2 1996 5796496262
## 3 1997 5873978244
## 4 1998 5950409046
## 5 1999 6026389348
## 6 2000 6102398541
```

```
# Next, aggregate the case count by year, gender and diagnosis and save in a dataframe
cases_by_year <- aggregate(who$count,
                           by=list(who$year,who$gender, who$diagnosis),
                           FUN = sum,
```

```
                           na.rm=TRUE)
colnames(cases_by_year) <- c("year","gender","diagnosis","cases")
head(cases_by_year)
```

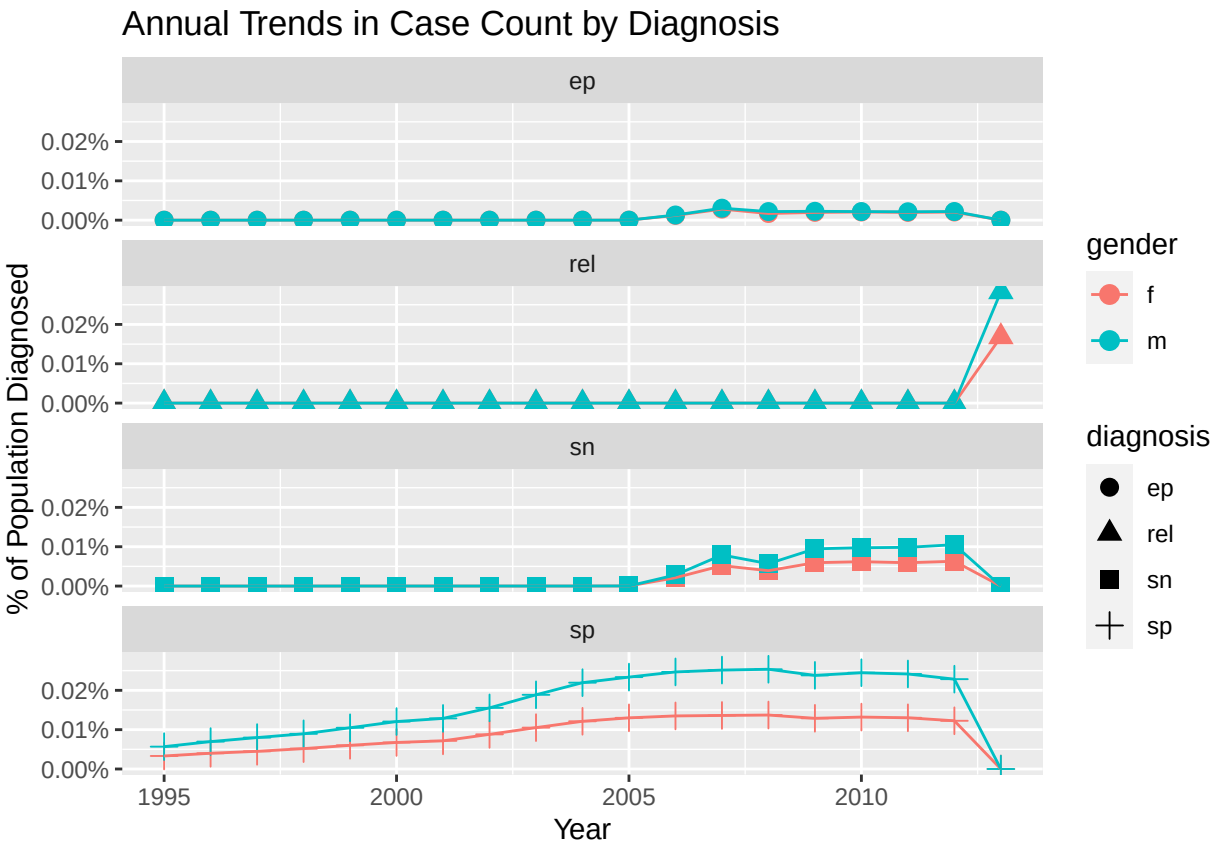
```
##   year gender diagnosis cases
## 1 1980      f        ep     0
## 2 1981      f        ep     0
## 3 1982      f        ep     0
## 4 1983      f        ep     0
## 5 1984      f        ep     0
## 6 1985      f        ep     0
```

```
# Now, combine the two dataframes together and eliminate the "NA" rows
cases_by_year <- left_join(cases_by_year,global_population,by="year") #combine
cases_by_year <- na.omit(cases_by_year) #eliminate nA
# Finally, add a column to the dataframe that calculates case counts as % of population
cases_by_year$percent_pop <-cases_by_year$cases/cases_by_year$population
head(cases_by_year)
```

```
##    year gender diagnosis cases population  percent_pop
## 16 1995      f        ep     0 5717507165 0.000000e+00
## 17 1996      f        ep     0 5796496262 0.000000e+00
## 18 1997      f        ep     0 5873978244 0.000000e+00
## 19 1998      f        ep     0 5950409046 0.000000e+00
## 20 1999      f        ep   126 6026389348 2.090804e-08
## 21 2000      f        ep   121 6102398541 1.982827e-08
```

8) Create a line plot in ggplot where your x axis contains the year and y axis contains the percent of world population. Facet this plot by diagnosis with each plot stacked vertically. You should have a line for each gender within each facet. Be sure to format your y axis and give your plot a title.

```
cases_by_year %>%
  ggplot(aes(x = year,
             y = percent_pop,
             color = gender,
             shape=diagnosis,
             group=interaction(gender,diagnosis))) +
  facet_wrap(~diagnosis,ncol=1) +
  geom_point(size=3) +
  geom_line() +
  xlab("Year") +
  ylab("% of Population Diagnosed") +
  ggtitle("Annual Trends in Case Count by Diagnosis") +
  scale_y_continuous(labels = scales::percent)
```



9) Now unite the min and max age variables into a new variable named age_range. Use a '-' as the separator.

```
# your code here
who$age_range <- paste(who$min_age,who$max_age,sep="-")
```

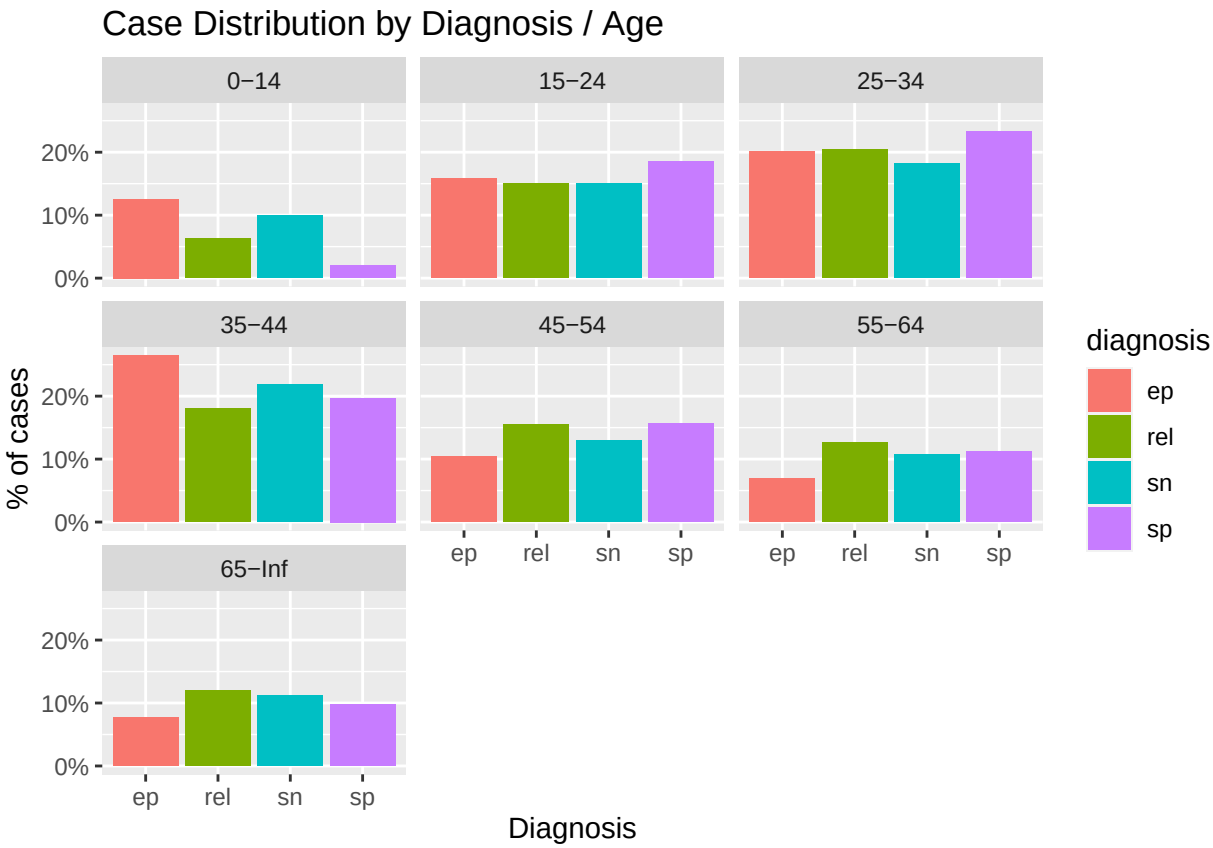
10) Find the percentage contribution of each age group by diagnosis. You will first need to find the count of all diagnoses then find the count of all diagnoses by age group. Join the former to the latter and calculate the percent of each age group. Plot these as a geom_col where the x axis is the diagnosis, y axis is the percent of total, and faceted by age group.

```
# First, count the total number of cases per diagnosis
count_diagnoses <- aggregate(who$count,
                             by=list(who$diagnosis),
                             FUN = sum,
                             na.rm=TRUE)
colnames(count_diagnoses) <- c("diagnosis","total_cases")

# Now, Further subdivide total cases by age group
diagnoses_by_age <- aggregate(who$count,
                              by=list(who$diagnosis,who$age_range),
                              FUN = sum,
                              na.rm=TRUE)
colnames(diagnoses_by_age) <- c("diagnosis","age_range","cases")

# Connect the two dataframes and calculate the percentage for each age group
diagnoses_by_age <- left_join(diagnoses_by_age,count_diagnoses,by="diagnosis")
diagnoses_by_age$percent_of_cases <- diagnoses_by_age$cases / diagnoses_by_age$total_cases
```

```
# Plot the results
diagnoses_by_age %>%
  ggplot(aes(x=diagnosis,
             y = percent_of_cases,
             fill=diagnosis)) +
  geom_col(show.legend = T) +
  facet_wrap(~age_range) +
  xlab("Diagnosis") +
  ylab("% of cases") +
  ggtitle("Case Distribution by Diagnosis / Age")+
  scale_y_continuous(labels = scales::percent)
```




```
# Obviously, if this were for a management presentation or something, I would need to dig
# into all the potential tweaks / arguments available within facet_wrap to improve the
# readability of this visualization.  While beyond the scope of this class, I tinkered and
# came up with the below which is better, but not perfect.  Lots to learn here!

diagnoses_by_age %>%
  ggplot(aes(x=diagnosis,
             y = percent_of_cases,
             fill=diagnosis)) +
  geom_bar(stat='identity',show.legend = T) +
  facet_grid(diagnosis~age_range) +
  xlab("Diagnosis") +
  ylab("% of cases") +
  ggtitle("Case Distribution by Diagnosis / Age")+
```

```
scale_y_continuous(labels = scales::percent)
```