Week 4 Exercises

Auriana Anderson

November 09, 2024

Please complete4 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)
```

```
## Warning: package 'dplyr' was built under R version 4.4.1
```

```
##
## 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
```

```r
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.4.1
```

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.4.1
```

```r
library(magrittr)
```

```
## Warning: package 'magrittr' was built under R version 4.4.1
```

```
##
## Attaching package: 'magrittr'
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

```r
data("who")

data("population")
```

```r
#code for this was in help for pivotlonger

who_new <- who %>% pivot_longer(
  cols = new_sp_m014:newrel_f65,
  names_to = c("diagnosis", "gender", "age"),
  names_pattern = "new_?(.*)_(.)(.*)",
  values_to = "count")

head(who_new)
```

```
## # A tibble: 6 x 8
##   country     iso2  iso3   year diagnosis gender age   count
##   <chr>       <chr> <chr> <dbl> <chr>     <chr>  <chr> <dbl>
## 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.

```r
who_new <- who_new %>%
left_join(population
          ,by = c("country", "year"))

head(who_new)
```

```
## # A tibble: 6 x 9
##   country     iso2  iso3   year diagnosis gender age   count population
##   <chr>       <chr> <chr> <dbl> <chr>     <chr>  <chr> <dbl>      <dbl>
## 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
```

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 problem.

```
who_new <- who_new%>%
separate(age,
         into = c("min_age", "max_age"),
         sep = 2,
         extra = "merge",
         remove = FALSE) %>%
  mutate(min_age = if_else(nchar(age) == 3, substr(age,1,1), min_age),
         max_age = if_else(nchar(age) == 3, substr(age,2,3), max_age))

head(who_new)
```

```
## # A tibble: 6 x 11
##   country     iso2  iso3   year diagnosis gender age   min_age max_age count
##   <chr>       <chr> <chr> <dbl> <chr>     <chr>  <chr> <chr>   <chr>   <dbl>
## 1 Afghanistan AF    AFG    1980 sp        m      014   0       14         NA
## 2 Afghanistan AF    AFG    1980 sp        m      1524  15      24         NA
## 3 Afghanistan AF    AFG    1980 sp        m      2534  25      34         NA
## 4 Afghanistan AF    AFG    1980 sp        m      3544  35      44         NA
## 5 Afghanistan AF    AFG    1980 sp        m      4554  45      54         NA
## 6 Afghanistan AF    AFG    1980 sp        m      5564  55      64         NA
## # i 1 more variable: population <dbl>
```

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.

```
# mine is currently is in the min age column so I will move into the max column

who_new <- who_new %>%
  mutate(min_age = if_else(age == "65" , "65", min_age),
         max_age = if_else(age == "65","Inf", max_age))

head(who_new, 8)
```

```
## # A tibble: 8 x 11
```

```
##   country     iso2  iso3   year diagnosis gender age   min_age max_age count
##   <chr>       <chr> <chr> <dbl> <chr>     <chr>  <chr> <chr>   <chr>   <dbl>
## 1 Afghanistan AF    AFG    1980 sp        m      014   0       14         NA
## 2 Afghanistan AF    AFG    1980 sp        m      1524  15      24         NA
## 3 Afghanistan AF    AFG    1980 sp        m      2534  25      34         NA
## 4 Afghanistan AF    AFG    1980 sp        m      3544  35      44         NA
## 5 Afghanistan AF    AFG    1980 sp        m      4554  45      54         NA
## 6 Afghanistan AF    AFG    1980 sp        m      5564  55      64         NA
## 7 Afghanistan AF    AFG    1980 sp        m      65    65      Inf        NA
## 8 Afghanistan AF    AFG    1980 sp        f      014   0       14         NA
## # i 1 more variable: population <dbl>
```

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

See ?sum for a hint on resolving NA values.

```
who_new_group <- who_new %>%
  group_by(gender,diagnosis) %>%
  summarise(count_per_diagnosis = sum(count,na.rm = TRUE))
```

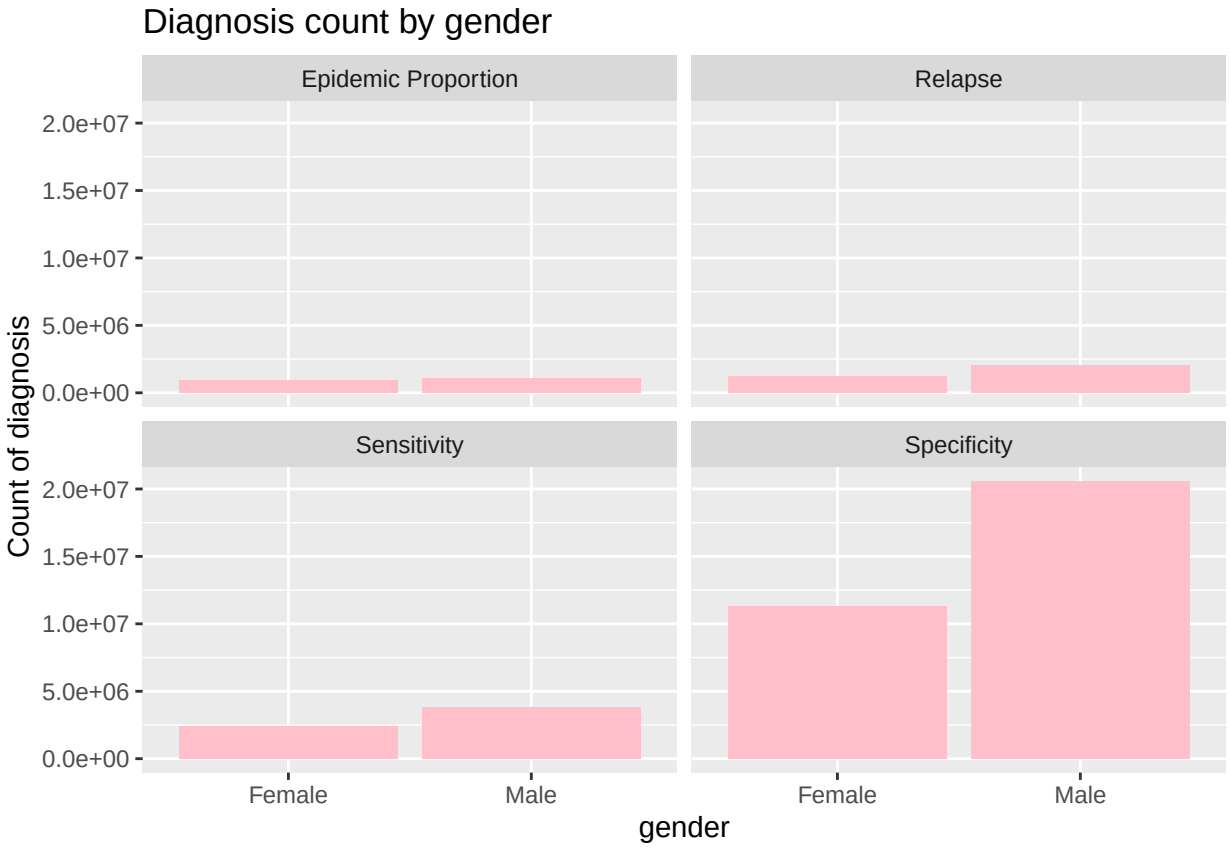
```
## `summarise()` has grouped output by 'gender'. You can override using the
## `.groups` argument.
```

```
head(who_new_group)
```

```
## # A tibble: 6 x 3
## # Groups:   gender [2]
##   gender diagnosis count_per_diagnosis
##   <chr>  <chr>                   <dbl>
## 1 f      ep                     941880
## 2 f      rel                   1201596
## 3 f      sn                    2439139
## 4 f      sp                   11324409
## 5 m      ep                    1044299
## 6 m      rel                   2018976
```

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.

```
ggplot(who_new_group,
       aes(x = gender,
       y = count_per_diagnosis)) +
  facet_wrap(~ diagnosis,
             labeller = labeller(
               diagnosis = c("ep" = "Epidemic Proportion",
                             "sp" = "Specificity",
                             "sn" = "Sensitivity",
                             "rel" = "Relapse"))) +
  geom_col(fill = "pink") +
  scale_x_discrete(labels = c("f" = "Female",
                              "m" = "Male")) +
  ylab("Count of diagnosis") +
  labs(title = "Diagnosis count by gender")
```



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

```
who_new_percentage <- who_new %>%
  group_by(year, gender, diagnosis) %>%
  filter(!is.na(count),!is.na(population)) %>%
  mutate(population_percentage = (count/population)*100)

head(who_new_percentage)
```

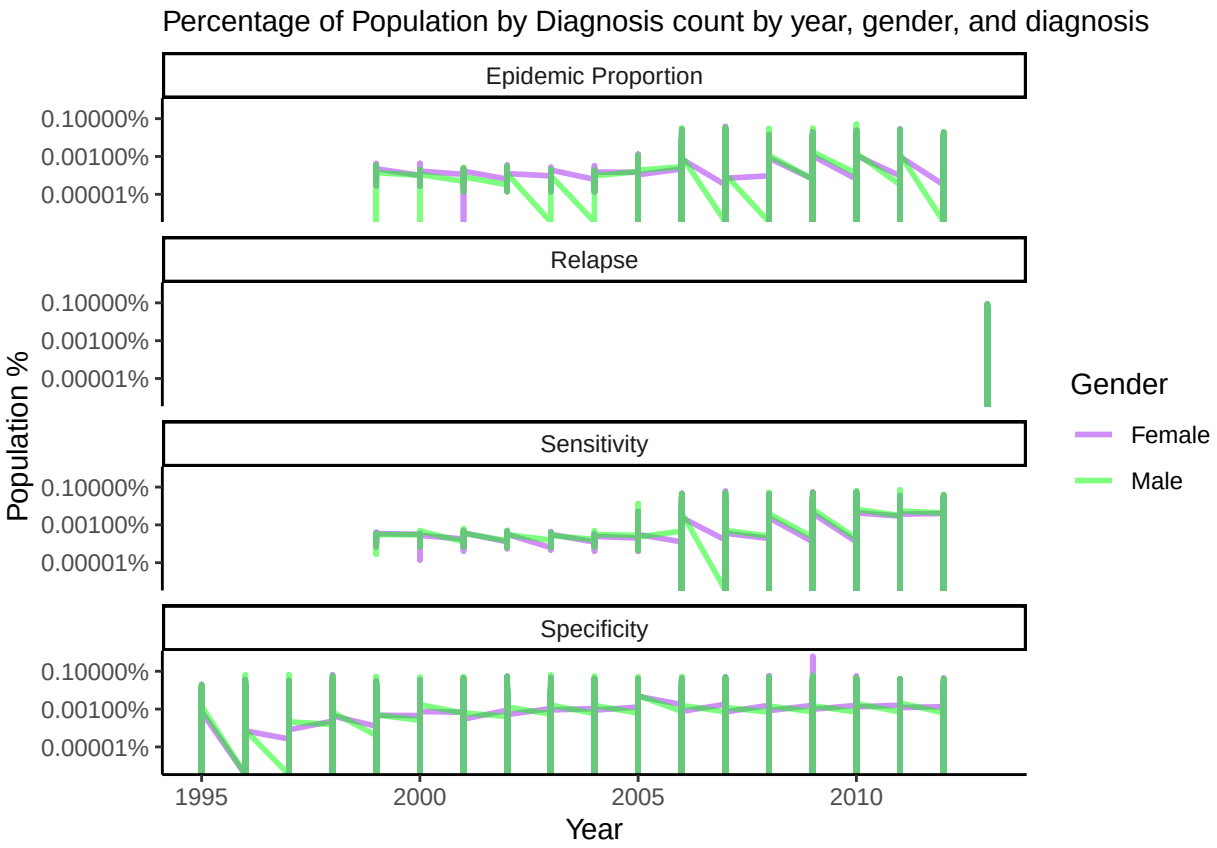
```
## # A tibble: 6 x 12
## # Groups:   year, gender, diagnosis [1]
##   country     iso2  iso3   year diagnosis gender age   min_age max_age count
##   <chr>       <chr> <chr> <dbl> <chr>     <chr>  <chr> <chr>   <chr>   <dbl>
## 1 Afghanistan AF    AFG    1997 sp        m      014   0       14          0
## 2 Afghanistan AF    AFG    1997 sp        m      1524  15      24         10
## 3 Afghanistan AF    AFG    1997 sp        m      2534  25      34          6
## 4 Afghanistan AF    AFG    1997 sp        m      3544  35      44          3
## 5 Afghanistan AF    AFG    1997 sp        m      4554  45      54          5
## 6 Afghanistan AF    AFG    1997 sp        m      5564  55      64          2
## # i 2 more variables: population <dbl>, population_percentage <dbl>
```

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.

```r
#chose a log10 scale for y for readability.
#I also thinks this works better for the different magnitudes of data
#not sure how I can make these more clear while faceted

ggplot(who_new_percentage,
       aes(x = year,
       y = population_percentage,
       colour = gender)) +
  facet_wrap(~ diagnosis, ncol = 1,
             labeller = labeller(
               diagnosis = c("ep" = "Epidemic Proportion",
                             "sp" = "Specificity",
                             "sn" = "Sensitivity",
                             "rel" = "Relapse"))) +
  geom_line(size = 1.0, alpha = 0.5) +
  scale_colour_manual(
    values = c("f" = "purple", "m" = "green"),
    labels = c("f" = "Female", "m" = "Male")) +
  labs(title = "Percentage of Population by Diagnosis count by year, gender, and diagnosis",
       x ="Year",
       y = "Population %",
       colour = "Gender") +
  scale_y_log10(labels = scales::percent_format(scale = 1)) +
  theme_classic() +
  theme(plot.title = element_text(size = 11))
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## Warning in scale_y_log10(labels = scales::percent_format(scale = 1)): log-10
## transformation introduced infinite values.
```



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

```
who_new <- who_new %>%
  unite(col= age_range, c(min_age, max_age), sep = "-")

head(who_new)
```

```
## # A tibble: 6 x 10
##   country    iso2  iso3   year diagnosis gender age   age_range count population
##   <chr>      <chr> <chr> <dbl> <chr>     <chr>  <chr> <chr>     <dbl>      <dbl>
## 1 Afghanist~ AF    AFG    1980 sp        m      014   0-14         NA         NA
## 2 Afghanist~ AF    AFG    1980 sp        m      1524  15-24        NA         NA
## 3 Afghanist~ AF    AFG    1980 sp        m      2534  25-34        NA         NA
## 4 Afghanist~ AF    AFG    1980 sp        m      3544  35-44        NA         NA
## 5 Afghanist~ AF    AFG    1980 sp        m      4554  45-54        NA         NA
## 6 Afghanist~ AF    AFG    1980 sp        m      5564  55-64        NA         NA
```

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 later 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.

```
count_diagnosis <- who_new %>%
  group_by(diagnosis) %>%
```

```
  summarise(count_of_diagnosis = sum(count, na.rm = TRUE)) %>%
  ungroup()

count_diagnosis_by_age <-who_new %>%
  group_by(diagnosis, age_range) %>%
  summarise(count_by_age = sum(count, na.rm = TRUE)) %>%
  ungroup()
```

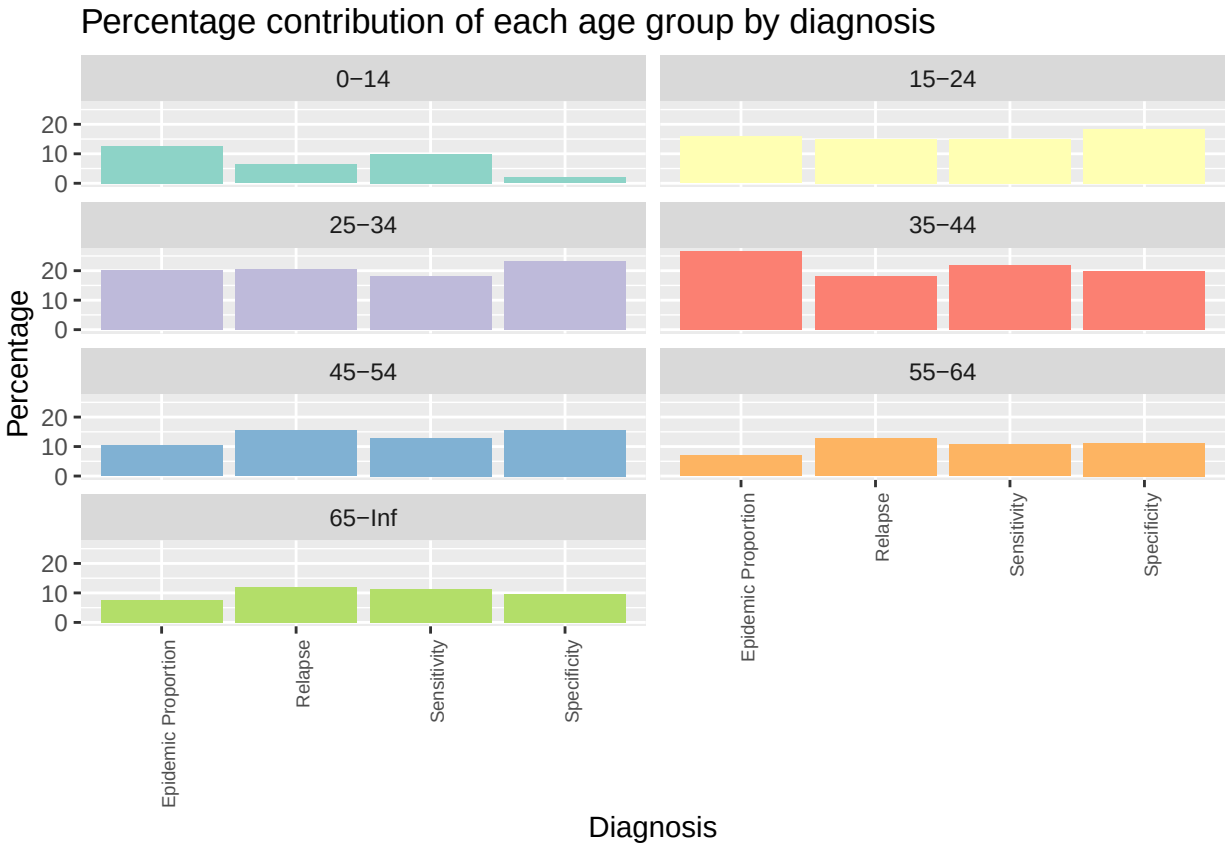
```
## `summarise()` has grouped output by 'diagnosis'. You can override using the
## `.groups` argument.
```

```
percentage_contribution <- count_diagnosis_by_age %>%
  left_join(count_diagnosis
            ,by = "diagnosis")

percentage_contribution <- percentage_contribution %>%
  mutate(percentage = (count_by_age/count_of_diagnosis)*100)


head(percentage_contribution)
```

```
## # A tibble: 6 x 5
##   diagnosis age_range count_by_age count_of_diagnosis percentage
##   <chr>     <chr>           <dbl>              <dbl>      <dbl>
## 1 ep        0-14           249998            1986179      12.6
## 2 ep        15-24          314716            1986179      15.8
## 3 ep        25-34          398758            1986179      20.1
## 4 ep        35-44          526041            1986179      26.5
## 5 ep        45-54          205633            1986179      10.4
## 6 ep        55-64          137356            1986179       6.92
```

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.

```
percentage_contribution %>%
ggplot(aes(x = diagnosis,
           y = percentage,
           fill = age_range)) +
  facet_wrap(~age_range, ncol = 2) +
  geom_col() +
  scale_fill_brewer(palette = "Set3") +
  labs(title = "Percentage contribution of each age group by diagnosis",
       x = "Diagnosis",
       y = "Percentage")+
  scale_x_discrete(labels = c("ep" = "Epidemic Proportion",
                              "sp" = "Specificity",
                              "sn" = "Sensitivity",
                              "rel" = "Relapse")) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size = 7),
    legend.position = "none")
```


Percentage contribution of each age group by diagnosis
0−14
15−24
25−34
35−44
45−54
55−64
65−Inf
Percentage
20
10
0
Epidemic Proportion
Relapse
Sensitivity
Specificity
Diagnosis