Milestone 3

Abigail Duque, Misty Garcia, Katherine Boudreau

2022-11-04

Milestone 1

Github link: https://github.com/r-public-health-group/monkey-pox

Team roles and responsibilities: https://github.com/r-public-health-group/monkey-pox/blob/main/team_roles_and_responsibilities.docx

Milestone 2

Description of dataset

What is the data source?

We are interested in how monkeypox case rates differ by European region and population demographics. We are first utilizing a monkeypox case dataset from the European Centre for Disease Prevention and Control that spans 2022-05-09 to 2022-08-23 and includes the daily number of confirmed monkey pox cases by country and how the data was collected. Then, we will incorporate a European population dataset from Eurostat, which includes European countries' yearly population data from 2011 to 2022 based on the total population residing in that country as of January 1st of each year. Finally, we will incorporate the European Statistical System's 2011 European census data which includes the EU country codes, sex at birth, age ranges, employment status (CAS), education status, population of locality of residence and the number of people in each strata.

How does the dataset relate to the group problem statement and question?

As we prepare our report on monkeypox case rates to the state health department, the monkeypox case data will provide us with the number of cases that will be the numerator of our case rate calculation, the population data gives us the size of the population at risk that will be the denominator of our case rate calculation, and the census data will allow us to stratify the case rates by various demographic factors.

Import Statement

All three of our CSVs are hosted on the course github page: https://github.com/PHW290/phw251_projectdata. We navigated to our respective csvs, clicked on raw data to get the standalone csv, and saved it into a url variable in this Rmd. We then used the read_csv function from the readr library to import our data into rstudio.

```
mpx_cases_url = "https://raw.githubusercontent.com/PHW290/phw251_projectdata/main/euro_mpx_cases.csv"
mpx_cases_df <- read_csv(mpx_cases_url)

pop_denoms_url = "https://raw.githubusercontent.com/PHW290/phw251_projectdata/main/euro_pop_denominators
pop_denoms_df = read_csv(pop_denoms_url)

census_stats_url = "https://raw.githubusercontent.com/PHW290/phw251_projectdata/main/euro_census_stats.
census_stats_df <- read_csv(census_stats_url,
          col_names = TRUE,
          col_types = NULL,
          na = c("", "NA"))
```

Lowercase Column Names

We are now lowercasing all the column names.

```
mpx_cases_df <- rename_with(mpx_cases_df, ~tolower(.x))
pop_denoms_df <- rename_with(pop_denoms_df, ~tolower(.x))
census_stats_df <- rename_with(census_stats_df, ~tolower(.x))
```

Data Types and Descriptions

Details of key data elements are outlined below. All elements listed are in an appropriate data format for the joins and analysis we intend to do.

Variable: daterep from mpx_cases_df

```
str(mpx_cases_df$daterep)
```

```
##  Date[1:2987], format: "2022-05-09" "2022-05-09" "2022-05-09" "2022-05-09" "2022-05-09" ...
```

```
min(mpx_cases_df$daterep)
```

```
## [1] "2022-05-09"
```

```
max(mpx_cases_df$daterep)
```

```
## [1] "2022-08-23"
```

```
unique(dplyr::count(mpx_cases_df, daterep)$n)
```

```
## [1] 29
```

The dates are in date format, they span from 2022-05-09 to 2022-08-23, and each date shows up 29 times.

Variables: countrycode from mpx_cases_df

```
typeof(mpx_cases_df$countrycode)
```

```
## [1] "character"
```

```
unique(mpx_cases_df$countrycode)
```

```
##  [1] "AT" "BE" "BG" "HR" "CY" "CZ" "DK" "EE" "FI" "FR" "DE" "EL" "HU" "IS" "IE"
## [16] "IT" "LV" "LT" "LU" "MT" "NL" "NO" "PL" "PT" "RO" "SK" "SI" "ES" "SE"
```

```
length(unique(mpx_cases_df$countrycode))
```

```
## [1] 29
```

The variable countryexp is the country name where countrycode is it's two letter counterpart. Both are in character format. There are 29 countries respresented.

Variable: source from mps_cases_df

```
typeof(mpx_cases_df$source)
```

```
## [1] "character"
```

```
unique(mpx_cases_df$source)
```

```
## [1] "TESSy" "EI"
```

Source is in a character format. There are two sources of data: "TESSy" and "EI."

Variable: confcase from mpx_cases_df

```
typeof(mpx_cases_df$confcases)
```

```
## [1] "double"
```

```
unique(mpx_cases_df$confcases)
```

```
##   [1]   0   1   4   2   3   6   5  20   9  11  82  21  14  13   7  10   8  16
##  [19]  22  26  12  19  15  27  30  33  47  25  31  18  41  34  40  38  61  43
##  [37]  42  72  46  48  66  70  55  56  95  97  53  23  90  76  89  86  24  28
```

```
##  [55]  67  17 101 110  74  93  83  58 106  71 130 151  79  68 109 144 184  69
##  [73]  29 176 191  78 158 102 199 112  37 131  44 113 155 160 140 655 136 170
##  [91]  60 172  45 159 388  63 139 114 137 284 147  51  77 121 124 128 178  91
## [109]  73  65
```

```
min(mpx_cases_df$confcases)
```

```
## [1] 0
```

```
mean(mpx_cases_df$confcases)
```

```
## [1] 5.715434
```

```
max(mpx_cases_df$confcases)
```

```
## [1] 655
```

The confirmed cases are in a numeric data format and range from 0 to 655. The mean is 5.7 cases per day.

Variable: geo from pop_denoms_df

```
typeof(pop_denoms_df$geo)
```

```
## [1] "character"
```

```
length(unique(pop_denoms_df$geo))
```

```
## [1] 54
```

The values in the GEO column of the pop_denoms_df are characters denoting the geopolitical region that a given row's population data comes from. There are 54 possible values for this column, either a 2-letter country code or a code denoting the entire European Union on a given year. We will eventually merge these 2-letter country codess with the 2-letter country codes in the euro_mpx_cases csv, so the character data type will suffice.

variable: obs_value from pop_denoms_df

```
summary(pop_denoms_df$obs_value)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##     31863   2088384   7040272  44417154  19702267 513206391
```

```
typeof(pop_denoms_df$obs_value)
```

```
## [1] "double"
```

The values in the obs_value column are the observed population on January 1 of the reported year. These values are doubles representing the total population count of a given region, so we can use these values directly as the denominators in our case rate calculations.

variable: country_code from census_stats_df

```
typeof(census_stats_df$country_code)
```

```
## [1] "character"
```

```
unique(census_stats_df$country_code)
```

```
##  [1] "AT" "BE" "BG" "CH" "CY" "CZ" "DE" "DK" "EE" "EL" "ES" "FI" "FR" "HR" "HU"
## [16] "IE" "IS" "IT" "LI" "LT" "LU" "LV" "MT" "NL" "NO" "PL" "PT" "RO" "SE" "SI"
```

```
## [31] "SK" "UK"
```

```
length(unique(census_stats_df$country_code))
```

```
## [1] 32
```

Country Code is a character data type and there are 32 different two-letter country codes in this dataset. We will obtain the different regions from the country_code variable.

variable: sex from census_stats_df

```
unique(census_stats_df$sex)
```

```
## [1] "F" "M"
```

Sex is a character data type, and there are two categories: male, female. We might use sex as a demographic variable to stratify case rates by.

variable: age from census_stats_df

```
typeof(census_stats_df$age)
```

```
## [1] "character"
```

```
unique(census_stats_df$age)
```

```
## [1] "Y_GE85" "Y_LT15" "Y15-29" "Y30-49" "Y50-64" "Y65-84"
```

Age is character data type that includes six different age ranges in the dataset: <15, 15-29, 30-49, 65-84 & >85. We might use age as a demographic variable to stratify case rates by.

variable: cas from census_stats_df

```
typeof(census_stats_df$cas)
```

```
## [1] "character"
```

```
unique(census_stats_df$cas)
```

```
## [1] "ACT"  "EMP"  "INAC" "UNE"  "UNK"
```

Cas is a character data type and represents the employment status of an individual. There are five different statuses, and we might use these to stratify case rates by.

variable: edu from census_stats_df

```
typeof(census_stats_df$edu)
```

```
## [1] "character"
```

```
unique(census_stats_df$edu)
```

```
## [1] "ED1"  "ED2"  "ED3"  "ED4"  "ED5"  "ED6"  "NAP"  "NONE" "UNK"
```

Education status is in a character format, and there are nine different statuses of education. We might use education status to stratify case rates by.

variable: res_pop from census_stats_df

```
typeof(census_stats_df$res_pop)
```

```
## [1] "character"
```

```
unique(census_stats_df$res_pop)
```

```
## [1] "500000-999999" "10000-99999"   "200000-499999" "100000-199999"
## [5] "GE1000000"     "1000-9999"     "0-1000"
```

Resident population (res_pop) is in a character data format. There are seven levels of population in each locality of residence.

variable: pop from census_stats_df

```
typeof(census_stats_df$pop)
```

```
## [1] "double"
```

```
range(census_stats_df$pop)
```

```
## [1]       0 1702270
```

The strata population is in a numeric data format. The number of people in the different location strata ranges from 0 to 1,702,270.

Milestone 3

Subset rows or columns, as needed

```
#acquire region dataset
regions <- read.csv('https://raw.githubusercontent.com/PHW290/phw251_projectdata/main/world_country_reg

#filtered by europe countries
regions <- regions %>%
  filter(region == 'Europe') %>%
  select(iso_3166.2, sub.region, region)

#create country code variable to join tables
regions <- regions %>%
  mutate(country_code = substr(regions$iso_3166.2,12,13) )

#filtered mpx_cases columns
mpx_cases_df <- mpx_cases_df %>%
  select(daterep, countrycode, confcases)

#filtered pop_denom by year and columns
pop_denoms_df <- pop_denoms_df %>%
  filter(time_period == 2022) %>%
  select(geo, obs_value)

#joined all three datasets together
df <- inner_join(x = regions, y = mpx_cases_df, by = c('country_code' = 'countrycode'))
df <- inner_join(x = df, y = pop_denoms_df, by = c('country_code' = 'geo'))


df <- df %>%
  select(-iso_3166.2, -region)
```

Create new variables needed for analysis (minimum 2)

```
#created case rate by ten million people
df_grouped <- df %>%
  group_by(sub.region, daterep) %>%
  summarise(total_cases = sum(confcases), total_obs = sum(obs_value)) %>%
  mutate(case_rate_by_tenmillion = total_cases / total_obs * 10000000)
```

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

```
df_grouped
```

```
## # A tibble: 412 x 5
## # Groups:   sub.region [4]
##    sub.region     daterep    total_cases total_obs case_rate_by_tenmillion
##    <chr>          <date>           <dbl>     <dbl>                   <dbl>
##  1 Eastern Europe 2022-05-09           0  89171711                       0
##  2 Eastern Europe 2022-05-13           0  89171711                       0
##  3 Eastern Europe 2022-05-15           0  89171711                       0
##  4 Eastern Europe 2022-05-16           0  89171711                       0
##  5 Eastern Europe 2022-05-17           0  89171711                       0
##  6 Eastern Europe 2022-05-18           0  89171711                       0
```

```
##  7 Eastern Europe 2022-05-19           0  89171711                      0
##  8 Eastern Europe 2022-05-20           0  89171711                      0
##  9 Eastern Europe 2022-05-21           0  89171711                      0
## 10 Eastern Europe 2022-05-22           0  89171711                      0
## # ... with 402 more rows
```

Clean variables needed for analysis (minimum 2)

```
#we used an inner join and there were no discrepancies between datasets,
#therefore are no nulls caused by joining
sum(is.na(df_grouped))
```

```
## [1] 0
```

```
#make sub.region a categorical value
df_grouped <- df_grouped %>%
  mutate(sub.region = factor(sub.region))

#make new month column & made it a categorical value
df_grouped <- df_grouped %>%
  mutate(month = strftime(daterep,"%B")) %>%
  mutate(month = factor(month, levels = c('May','June','July','August')))

str(df_grouped)
```

```
## grouped_df [412 x 6] (S3: grouped_df/tbl_df/tbl/data.frame)
##  $ sub.region             : Factor w/ 4 levels "Eastern Europe",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ daterep                : Date[1:412], format: "2022-05-09" "2022-05-13" ...
##  $ total_cases            : num [1:412] 0 0 0 0 0 0 0 0 0 0 ...
##  $ total_obs              : num [1:412] 89171711 89171711 89171711 89171711 89171711 ...
##  $ case_rate_by_tenmillion: num [1:412] 0 0 0 0 0 0 0 0 0 0 ...
##  $ month                  : Factor w/ 4 levels "May","June","July",..: 1 1 1 1 1 1 1 1 1 1 ...
##  - attr(*, "groups")= tibble [4 x 2] (S3: tbl_df/tbl/data.frame)
##   ..$ sub.region: Factor w/ 4 levels "Eastern Europe",..: 1 2 3 4
##   ..$ .rows     : list<int> [1:4]
##   .. ..$ : int [1:103] 1 2 3 4 5 6 7 8 9 10 ...
##   .. ..$ : int [1:103] 104 105 106 107 108 109 110 111 112 113 ...
##   .. ..$ : int [1:103] 207 208 209 210 211 212 213 214 215 216 ...
##   .. ..$ : int [1:103] 310 311 312 313 314 315 316 317 318 319 ...
##   .. ..@ ptype: int(0)
##   ..- attr(*, ".drop")= logi TRUE
```

Data dictionary based on clean dataset (minimum 4 data elements), including: Fields Variable name Data type Description

sub.region: factor, region in europe

daterep: date, year-month-day by week from may to august 2022

total_cases: num, total monkeypox cases in a subregion by week

total_obs: num, total population by subregion

case_rate_by_tenmillion: num, the rate of monkeypox in a subregion by ten million people

month: date, the month from may to august 2022

One or more tables with descriptive statistics for 4 data element

```
summary(df_grouped)
```

```
##            sub.region     daterep            total_cases      total_obs
##  Eastern Europe :103   Min.   :2022-05-09   Min.   :  0.0   Min.   : 38749061
##  Northern Europe:103   1st Qu.:2022-06-07   1st Qu.:  1.0   1st Qu.: 76566048
##  Southern Europe:103   Median :2022-07-03   Median :  8.0   Median :106223452
##  Western Europe :103   Mean   :2022-07-02   Mean   : 41.3   Mean   :110280452
##                        3rd Qu.:2022-07-29   3rd Qu.: 53.0   3rd Qu.:139937856
##                        Max.   :2022-08-23   Max.   :764.0   Max.   :189925840
##  case_rate_by_tenmillion    month
##  Min.   : 0.000          May   : 76
##  1st Qu.: 0.158          June  :120
##  Median : 1.009          July  :124
##  Mean   : 3.048          August: 92
##  3rd Qu.: 3.894
##  Max.   :40.226
```