GY7702-Assignment-1

Robert Atkinson

10/11/2020

Document was written in RMarkdown with a PDF as output.

The repository used to make this document can be found at https://github.com/RobertSA92/GY7702-Assignment-1.

Question 1

```
# Create a vector of  25 numbers between 1 and 7 using the function c.

survey_answers <- c(NA, 3, 4, 4, 5, 2, 4, NA, 6, 3, 5, 4, 0, 5,
                    7, 5, NA, 5, 2, 4, NA, 3, 3, 5, NA)



# These values represent the answers to a survey question, the scale is as follows:

# 1 = completely disagree
# 2 = disagree
# 3 = somehow disagree
# 4 = neither agree nor disagree
# 5 = somehow agree
# 6 = agree
# 7 = completely agree
# NA = missing value
```

Question 1.1:

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.1     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(knitr)


# Remove all NA values from vector
 cleaned_survey_answers <- survey_answers[!is.na(survey_answers)]

    # Check if all survey answers completely disagree
    if(all(cleaned_survey_answers == 1)) {
      cat("All survey answers completely disagree")


    # Check if all survey answers compleely agree
    } else {
    if(all(cleaned_survey_answers == 7)) {
      cat("All survey answers completely agree")


    # If all answers neither completely disagree or completeyly agree give this output
    } else {
      cat("All survey answers neither completely disagree or completely agree")
  }
}
```

```
## All survey answers neither completely disagree or completely agree
```

Question 1.2:

```r
# Coerce data into logical values, any values that equal 5 or greater are TRUE,
# any values that equal 4 or lower are FALSE.
# This uses the original data with the NA values in order to find the original index
# of each survey answer.

 logical_original_survey_answers <- survey_answers >= 5


# Find the indexes that are TRUE

  which(logical_original_survey_answers %in% TRUE)
```

```
## [1]  5  9 11 14 15 16 18 24
```

Question 2

Question 2.1:

```r
# Install the library library palmerpenguins.

  install.packages("palmerpenguins", repos = "https://cran.r-project.org/")
```

```
## Installing package into '/home/sgge28/R/x86_64-pc-linux-gnu-library/4.0'
## (as 'lib' is unspecified)
```

```
  library(palmerpenguins)
```

Question 2.2:

```
# Create identifier to store table and load penguins date from palmerpenguins.
  high_body_mass_gentoo <- palmerpenguins::penguins %>%

  # Select data for the species, island, bill length (mm), and body mass (g)
    dplyr::select(species, island, bill_length_mm, body_mass_g) %>%

  # Filter for penguins that are the Gentoo species.
    dplyr::filter(species == "Gentoo") %>%

  # Remove observations that have a NA value for bill length.
    dplyr::filter(!is.na(bill_length_mm)) %>%

  # Arrange data to have the highest body mass (g) at the top.
    dplyr::arrange(desc(body_mass_g)) %>%

  # Create tibble from data.
  tibble::as_tibble()



  # Show a well-formated table.
   high_body_mass_gentoo %>%

  # Only show the top 10 rows, reflecting the top 10 highest values for body mass (g).
   dplyr::slice_head(n = 5) %>%
   knitr::kable() %>%
   print()
```

```
##
##
## |species |island | bill_length_mm| body_mass_g|
## |:-------|:------|--------------:|-----------:|
## |Gentoo  |Biscoe |           49.2|        6300|
## |Gentoo  |Biscoe |           59.6|        6050|
## |Gentoo  |Biscoe |           51.1|        6000|
## |Gentoo  |Biscoe |           48.8|        6000|
## |Gentoo  |Biscoe |           45.2|        5950|
```

Question 2.3:

```
 # Create identifier to store table and load penguins date from palmerpenguins.
 average_bill_penguins <- palmerpenguins::penguins %>%

  # Select data for island, bill length (mm), and body mass (g).
   dplyr::select(island, bill_length_mm, body_mass_g) %>%

  # Remove observations that have a NA value for bill length.
```

```
   dplyr::filter(!is.na(bill_length_mm)) %>%

  # Arrange bill length (mm) values in descending order.
   dplyr::arrange(desc(bill_length_mm)) %>%

  # Group data by island.
   dplyr::group_by(island) %>%

  # Find the mean average of bill length (mm) for each island.
   dplyr::summarise(
     mean(bill_length_mm)
   ) %>%

  # Create tibble from data
    tibble::as_tibble()

  # Show a well-formated table.
  average_bill_penguins %>%
  knitr::kable() %>%
  print()
```

```
##
##
## |island    | mean(bill_length_mm)|
## |:---------|--------------------:|
## |Biscoe    |             45.25749|
## |Dream     |             44.16774|
## |Torgersen |             38.95098|
```

Question 2.4:

```
# Create identifier to store table and load penguins date from palmerpenguins.
 min_med_max_bill_proportion_penguins <- palmerpenguins::penguins %>%

  # Select data for species, island, bill length (mm), and bill depth (mm).
   dplyr::select(species, island, bill_length_mm, bill_depth_mm) %>%

  # Remove observations that have a NA value for bill length.
   dplyr::filter(!is.na(bill_length_mm)) %>%

  # Group by species
   dplyr::group_by(species) %>%

  # Create new variable containing the propotions between bill length
  # and bil depth for each penguin.
   dplyr::mutate(
     proportion_between_bl_and_bd = (((bill_length_mm - bill_depth_mm)
                                      /bill_length_mm)*100)
   ) %>%

  # Summarise proportion by species.
  summarise(
```

```
    # Summarise the mininum proportion.
     min_proportion = round(min(proportion_between_bl_and_bd), 2),

    # Summarise the median proportion.
     median_proportion = round(median(proportion_between_bl_and_bd), 2),

    # Summarise the maximum proportion.
     max_proportion = round(max(proportion_between_bl_and_bd), 2)

         ) %>%

    # Create tibble from data
    tibble::as_tibble()


    # Show a well-formatted table.
    min_med_max_bill_proportion_penguins %>%
    knitr::kable() %>%
    print()
```

```
##
##
## |species   | min_proportion| median_proportion| max_proportion|
## |:---------|--------------:|-----------------:|--------------:|
## |Adelie    |          39.02|             53.20|          59.18|
## |Chinstrap |          57.46|             62.43|          69.31|
## |Gentoo    |          61.04|             68.42|          72.32|
```

Question 3

Question 3.1:

```
#  Load the data from covid19_cases_20200301_20201017.csv to a variable named covid_data.
 covid_data <- readr::read_csv("covid19_cases_20200301_20201017.csv")
```

```
## Parsed with column specification:
## cols(
##   specimen_date = col_date(format = ""),
##   area_name = col_character(),
##   newCasesBySpecimenDate = col_double(),
##   cumCasesBySpecimenDate = col_double()
## )
```

Question 3.2:

```
# Store resulting table in variable.
powys_complete_covid_data <- covid_data %>%

  # Replace NA values with the value available for the previous date.
  tidyr::fill(newCasesBySpecimenDate, .direction = "down") %>%
```

```
  # Replace the remaining NA values with zero.
  dplyr::mutate(newCasesBySpecimenDate = replace_na(newCasesBySpecimenDate, 0)) %>%

  # Subset only data for the area Powys.
  dplyr::filter(area_name == "Powys") %>%

  # Drop the area_name column.
  dplyr::select(-area_name) %>%

  # Create tibble from data
  tibble::as_tibble()


# Show a well-formatted table. Limit to only show the first 5 rows.
powys_complete_covid_data %>%
  dplyr::slice_head(n = 5) %>%
  knitr::kable() %>%
  print()
```

```
##
##
## |specimen_date | newCasesBySpecimenDate| cumCasesBySpecimenDate|
## |:-------------|----------------------:|----------------------:|
## |2020-03-01    |                      0|                      0|
## |2020-03-02    |                      0|                      0|
## |2020-03-03    |                      0|                      0|
## |2020-03-04    |                      0|                      0|
## |2020-03-05    |                      0|                      0|
```

Question 3.3:

```
# Create a copy of "powys_complete_covid_data".
powys_day_before <- powys_complete_covid_data

# Load the lubridate library.
 library(lubridate)

# Assign new information to the "powys_day_before" variable.
 powys_day_before <- powys_day_before %>%

   # Create new column named "day_before" that reports the day before the day reported
   # in the column specimen_date
   dplyr::mutate(day_before = ymd(specimen_date) - 1) %>%

   # Coerce "day_before" data to a character value.
   dplyr::mutate(day_before = as.character(day_before)) %>%

   # Drop the "specimen_date" and "cumCasesesBySpecimenDate columns".
   dplyr::select(-specimen_date, -cumCasesBySpecimenDate) %>%

   # Rename the "newCasesBySpecimenDate" column to "newCases_day_before.
   dplyr::rename("newCases_day_before" = "newCasesBySpecimenDate") %>%
```

```
  # Create tibble from data
  tibble::as_tibble()


# Join "powys_day_before" with "powys_complete_covid_data". Match the "specimen_date"
# and "day_before" columns.


# Coerce "specimen_date" to characters values in order for the data type to match
# the "day_before" data.
powys_complete_covid_data <- powys_complete_covid_data %>%
 dplyr::mutate(specimen_date = as.character(specimen_date))


# Store resulting table in a varable named "powys_covid_development".
powys_covid_development <-

  # Use inner join verb.
  dplyr::inner_join(

  # Left table.
  powys_complete_covid_data,

  # Right table.
  powys_day_before,

  # Columns to match.
  by = c( "specimen_date" ="day_before")
) %>%


  # Reorder columns for clearer reading, drop the "cumCasesBySpecimenDate" column.
  dplyr::select(specimen_date, newCases_day_before, newCasesBySpecimenDate,
                -cumCasesBySpecimenDate) %>%


  # Calculate a new column in the joined table.

  # This joined table will contain the number of new cases as a percentage of
  # the number of new cases of the day before.

  dplyr::mutate(
    percentage = (newCasesBySpecimenDate/ newCases_day_before)*100
    ) %>%

  # Create tibble from data.
  tibble::as_tibble()


# Show a well-formatted table.

powys_covid_development %>%
```

```
  # Limit to only show the first 5 rows.

   dplyr::slice_head(n = 5) %>%

  # Show only the first 2 decimal places.
   knitr::kable(digits = 2)
```

specimen_date	newCases_day_before	newCasesBySpecimenDate	percentage
2020-03-01	0	0	NaN
2020-03-02	0	0	NaN
2020-03-03	0	0	NaN
2020-03-04	0	0	NaN
2020-03-05	0	0	NaN

Question 3.4:

The number of new cases starts to rise in the second week of March but quickly lowers back to 0 after 4 days. An increase can be observed later in March, this time with sustained percentage changes often between 100 - 200%. The lower figures that interrupt this sustained change tend to be weekend recordings, for example, March 28th and 29th. The growth of new cases continues into May, reaching a percentage change high of 800% on May 10th. From here, the number of new cases declines. In June, July, and the first week of August there are very few new cases, however, there is a notable spike of 10 new cases on July 20th. New cases begin to climb again in the 2nd half of August, with percentage change values frequently between 100 - 300%. The figures remain similar to this through to the last recording on October 9th.

Question 4

Table 1.

```
# Load population data into the variable "lad19".

lad19 <- readr::read_csv("lad19_population.csv")


# Join lad19 information with the information from
# the file covid19_cases_20200301_20201017.csv
# found in the variable "covid_data" from Question 3.
# Match columns ""lad19_area_name" from "lad19", and "area_name" from "covid_data".


lad19_covid19 <-

  # Use inner join verb.
  dplyr::inner_join(

  # Left table.
       lad19,

  # Right table.
```

```
    covid_data,

  # Columns to match.
    by = c("lad19_area_name" = "area_name")
   )


# Finding areas with similar population to Powys. Create Table 1.

# Assign the areas with a similar population to Powys to the "selected_areas" variable.
 selected_areas <- lad19 %>%

   # Drop the "lad19_area_code" column.
   select(-lad19_area_code) %>%

   # Powys has a population of 132531. Filter for a population 1000 above this
   # and 1000 below.
   filter(area_population %in% (131531:133531)) %>%

   # Arrange by area population.
   arrange(area_population) %>%

   # Rename "lad19_area_name" to "area_name".
   rename(area_name = lad19_area_name) %>%

   # Create a tibble from this data
   tibble::as_tibble()

   # Show a well-formated table
 selected_areas %>%
   knitr::kable()
```

area_name	area_population
Powys	132531
Epping Forest	133334
Exeter	133334
Stratford-on-Avon	133334
Tonbridge and Malling	133334

Table 2.

```
# Finding the 7 day average of "Powys", "Epping Forest", "Exeter", "Straford-on-Avon",
# and "Tonbridge and Malling". Create Table 2.

# Assign table to the variable "seven_day_average_5_areas".
seven_day_average_5_areas <- lad19_covid19 %>%

  # Drop the columns "cumCasesBySpecimenDate", "area_population", and "lad19_area_code".
  select(-cumCasesBySpecimenDate, -area_population, -lad19_area_code) %>%

  # Change spaces and hyphens to underscores in order to work in R.
```

```
  dplyr::mutate(
    lad19_area_name = str_replace_all(lad19_area_name, " ", "_"),
    lad19_area_name = str_replace_all(lad19_area_name, "-", "_")
  ) %>%

 # Filter for the areas "Powys", "Epping Forest", "Exeter", "Stratford-on-Avon",
 # and "Tonbridge and Malling".

 filter(lad19_area_name == "Powys"|
          lad19_area_name == "Epping_Forest"|
          lad19_area_name == "Exeter"|
          lad19_area_name == "Stratford_on_Avon" |
          lad19_area_name == "Tonbridge_and_Malling") %>%

 # Pivot table to wide format.
 pivot_wider(

  # Retrieve colum names from "lad19_area_name".
 names_from = lad19_area_name,

 #  Retrieve values from "newCasesBySpecimenDate".
 values_from = newCasesBySpecimenDate
 ) %>%

 # Find the start date for each week. Class the start date as Monday.
 # Use this to create new column called "week_commencing".
 mutate(
   week_commencing = floor_date(specimen_date, unit = "weeks",
                                week_start = getOption("lubridate.week.start", 1))
 ) %>%

 # Group data by each "week_commencing" date, condesing the rows to weeks.
 group_by(week_commencing) %>%

 # Find the 7-day average of new cases for each selected area.
 summarise(Powys = mean(Powys),
           Epping_Forest = mean(Epping_Forest),
           Exeter = mean(Exeter),
           Stratford_on_Avon = mean(Stratford_on_Avon),
           Tonbridge_and_Malling = mean(Tonbridge_and_Malling)) %>%

# Check if there are any NA values, if so, replace with previous date value.

 tidyr::fill(Powys, .direction = "down") %>%
 tidyr::fill(Epping_Forest, .direction = "down") %>%
 tidyr::fill(Exeter, .direction = "down") %>%
 tidyr::fill(Stratford_on_Avon, .direction = "down") %>%
 tidyr::fill(Tonbridge_and_Malling, .direction = "down") %>%

# If there are still NA values replace them with 0.
dplyr::mutate(

   Powys = replace_na(Powys, 0),
```

```
    Epping_Forest = replace_na(Epping_Forest, 0),
    Exeter = replace_na(Exeter, 0),
    Stratford_on_Avon = replace_na(Stratford_on_Avon, 0),
    Tonbridge_and_Malling = replace_na(Tonbridge_and_Malling, 0)) %>%

 # Create tibble from data.
  tibble::as_tibble()



# Show data in a well-formatted table.

seven_day_average_5_areas %>%

  # Limit to only show the first 5 rows.

  dplyr::slice_head(n = 5) %>%

  # Show only the first 2 decimal places.

  knitr::kable(digits = 2)
```

week_commencing	Powys	Epping_Forest	Exeter	Stratford_on_Avon	Tonbridge_and_Malling
2020-02-24	0.00	0.00	0.00	0.00	0.00
2020-03-02	0.00	0.00	0.29	0.00	0.00
2020-03-09	1.29	0.86	0.29	0.00	0.43
2020-03-16	0.29	3.14	0.57	2.29	1.14
2020-03-23	2.00	9.29	2.14	4.86	4.00

The aim of this analysis was to use official government data to compare the development of Covid-19 cases in Powys with other areas in the United Kingdom that have a similar population count. According to Local Authority Data the area of Powys has a population of 132531. To find other regions with a similar population, the data was filtered for areas with a population between +1000 and -1000 that of Powys. The areas found by this process were Epping Forest, Exeter, Stratford-on-Avon, and Tonbridge and Malling (Table 1.)

Data recording the daily number of new Covid cases was then analysed for each area. The results showed a fall in cases on weekend days, to mitigate this weekend factor, and attain a better understanding of general trends, a 7-day mean average of new cases was calculated for each week.

The first 8 weeks show the number of new cases in Powys increasing, reaching a high of 6.71 in mid-April. The 4 other areas show a similar pattern, reaching their highest number of new cases either in April, or at the very end of March. From here, all areas begin to decline, Powys reaches a low of 0.14 at the start of June and sustains a value of less than 1 every week until the end of July. Similarly, all other areas reach low values in July or the latter half of June. The number of new cases then grows in every area until the data ends in October (Table 2).