Project 1

Jason Ballinger, Jason Perez, Megan Juza

2024-04-07

Contents

Background

The World Health Organization has recently employed a new data science initiative, *CSIT-165*, that uses data science to characterize pandemic diseases. *CSIT-165* disseminates data driven analyses to global decision makers.

CSIT-165 is a conglomerate comprised of two fabricated entities: *Global Health Union (GHU)* and *Private Diagnostic Laboratories (PDL)*. Your and your partner's role is to play a data scientist from one of these two entities.

Data

2019 Novel Coronavirus COVID-19 (2019-nCoV) Data Repository by John Hopkins CSSE

Data for 2019 Novel Coronavirus is operated by the John Hopkins University Center for Systems Science and Engineering (JHU CSSE). Data includes daily time series CSV summary tables, including confirmations, recoveries, and deaths. Country/region are countries/regions hat conform to World Health Organization (WHO). Lat and Long refer to coordinates references for the user. Date fields are stored in MM/DD/YYYY format.

Project Objectives

Objective 1

```r
# Objective 1: Determine where COVID-19 originated from

library(readr)

# Import data sets
cases <- data.frame(read_delim(file="time_series_covid19_confirmed_global.csv", delim=","))
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
deaths <- data.frame(read_delim(file="time_series_covid19_deaths_global.csv", delim=","))
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# Highest confirmed cases & deaths for first day (1/22/2020)
highestCasesIndex <- which.max(cases$X1.22.20)
highestDeathsIndex <- which.max(cases$X1.22.20)

# Get location given the index
highestCasesLocation <- paste(cases$Province.State[highestCasesIndex], cases$Country.Region[highestCases
highestDeathsLocation <- paste(deaths$Province.State[highestDeathsIndex], cases$Country.Region[highestDe

# Check if the locations are the same
if (highestCasesLocation == highestDeathsLocation) {
  print(paste("COVID-19 originated from:", highestCasesLocation))
} else {
  print("Unable to find COVID-19's origin location")
}
```

```
## [1] "COVID-19 originated from: Hubei China"
```

Objective 2

```
# Objective 2: Where is the most recent area to have a first confirmed case?

library(readr)

# Import data set
cases <- data.frame(read_delim(file="time_series_covid19_confirmed_global.csv", delim=","))
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Find the indices of the most recent first case
recentRow <- NA
recentCol <- NA

for (i in 1:nrow(cases)) {
  row <- cases[i, ] # Splice the row (state/country)
  for (j in 5:ncol(cases)) {
    # Check for first case
    if (cases[i, j] >= 1) {
      # Check if it has a more recent first case
      if (j > recentCol || is.na(recentCol)) {
        recentRow <- i
        recentCol <- j
      }
      break
    }
  }
}

# return the state + country value for the highest index
print(paste("Region with the newest first case is:", cases[recentRow, 1], cases[recentRow, 2]))
```

```
## [1] "Region with the newest first case is: Pitcairn Islands United Kingdom"
```

Objective 3

```r
# Objective 3: How far away are the areas from objective 2 from where the first confirmed case(s) occur

library(geosphere)

# Coordinates of the most recent area to have a first confirmed case
recent_coords <- c(cases[recentRow, "Long"], cases[recentRow, "Lat"])

# Coordinates of the area with the highest number of confirmations and deaths on the first recorded day
origin_coords <- c(cases[highestCasesIndex, "Long"], cases[highestCasesIndex, "Lat"])

# Calculate distance in meters
distance_meters <- distm(origin_coords, recent_coords, fun = distVincentyEllipsoid)

# Convert meters to miles (1 meter = 0.000621371 miles)
distance_miles <- distance_meters * 0.000621371

# Print the result
cat(paste("The region with the newest first case, ", cases[recentRow, "Province.State"], ", ", cases[re
          ", is approximately ", round(distance_miles, 2), " miles away from ", highestCasesLocation, "
```

```
## The region with the newest first case, Pitcairn Islands, United Kingdom, is approximately 8746.97 mi
```

Objective 4

```r
# Objective 4: Calculate risk scores for different areas

# Filter out areas with zero confirmations
cases_filtered <- cases[cases$X1.22.20 > 0, ]
```

```
deaths_filtered <- deaths[deaths$X1.22.20 > 0, ]

# Calculate risk scores
risk_scores <- 100 * deaths_filtered$X1.22.20 / cases_filtered$X1.22.20

# Add risk scores to the dataset
cases_filtered$risk_score <- risk_scores

# Sort areas by risk score and confirmations
sorted_areas <- cases_filtered[order(cases_filtered$risk_score, -cases_filtered$X1.22.20), ]

# Get the area with the lowest risk score and most confirmations
lowest_risk_area <- head(sorted_areas, 1)

# Get the area with the highest risk score and most confirmations
highest_risk_area <- tail(sorted_areas, 1)

# Calculate global risk score
global_risk_score <- 100 * sum(deaths_filtered$X1.22.20) / sum(cases_filtered$X1.22.20)

# Print the results using cat(paste())
cat(paste("Lowest risk score:", lowest_risk_area$risk_score))
```

```
## Lowest risk score: 3.82882882882883
```

```
cat("\n") # Newline for readability
```

```
cat(paste("Highest risk score:", highest_risk_area$risk_score))
```

```
## Highest risk score: 1700
```

```
cat("\n") # Newline for readability
```

```
cat(paste("Global risk score:", global_risk_score))
```

```
## Global risk score: 3.05206463195691
```

Objective 5

```
# Objective 5: Make two tables with the top 5 countries that have the most COVID-19 related confirmatio

library(readr)

# import data sets
confirmed <- data.frame(read_delim(file="time_series_covid19_confirmed_global.csv", delim=","))
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
deaths <- data.frame(read_delim(file="time_series_covid19_deaths_global.csv", delim=","))
```

```
## Rows: 289 Columns: 1147
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr    (2): Province/State, Country/Region
## dbl (1145): Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20, 1/26/20, 1/27/20,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# list of countries in the dataframe
countries <- unique(confirmed$Country.Region)

# create empty lists to store summed confirmed and deaths counts
summed_confirmed <- vector("numeric", length(countries))
summed_deaths <- vector("numeric", length(countries))

# sum confirmed and death counts for each country
for (i in 1:length(countries)) {
  country <- countries[i]
  summed_confirmed[i] <- sum(confirmed[confirmed$Country.Region == country, c(5:length(confirmed))])
  summed_deaths[i] <- sum(deaths[deaths$Country.Region == country, c(5:length(deaths))])
}

# create data frames with summed counts and countries
sum_confirmed_dataframe <- data.frame(Country = countries, Confirmed = summed_confirmed)
sum_deaths_dataframe <- data.frame(Country = countries, Deaths = summed_deaths)

# put countries in data frames in descending order based on counts
sum_confirmed_dataframe <- sum_confirmed_dataframe[order(-sum_confirmed_dataframe$Confirmed), ]
sum_deaths_dataframe <- sum_deaths_dataframe[order(-sum_deaths_dataframe$Deaths), ]

# use kable to convert data frames into tables
library(knitr)
confirmed_table <- kable(sum_confirmed_dataframe[1:5, ], caption = "Top 5 Countries with the Most COVID-
deaths_table <- kable(sum_deaths_dataframe[1:5, ], caption = "Top 5 Countries with the Most COVID-19 Dea

# print tables
print(confirmed_table)
```

```
##
##
## Table: Top 5 Countries with the Most COVID-19 Confirmations
##
## |    |Country |   Confirmed|
## |:---|:-------|-----------:|
## |187 |US      | 53813184406|
## |81  |India   | 29131119694|
## |25  |Brazil  | 21182690594|
## |64  |France  | 16105911886|
## |68  |Germany | 13686043720|
```

```r
print(deaths_table)
```

```
##
##
## Table: Top 5 Countries with the Most COVID-19 Deaths
##
## |    |Country |    Deaths|
## |:---|:-------|---------:|
## |187 |US      | 713877215|
## |25  |Brazil  | 488181000|
## |81  |India   | 364921237|
## |118 |Mexico  | 241085189|
## |148 |Russia  | 220983590|
```

GitHub Log

```
git log --pretty=format:"%nSubject: %s%nAuthor: %aN%nDate: %aD%nBody: %b"
```

```
##
## Subject: Objectives 3 & 4 completed
## Author: jperez4911
## Date: Sun, 7 Apr 2024 20:17:34 -0700
## Body:
##
## Subject: writeup updated with objective 5
## Author: Megan Juza
## Date: Sun, 7 Apr 2024 18:06:31 -0700
## Body:
##
## Subject: added objective 5 in the writup and created R script document, objective5
## Author: Megan Juza
## Date: Sun, 7 Apr 2024 17:43:17 -0700
## Body:
##
## Subject: objective 2
## Author: Jason Ballinger
## Date: Wed, 3 Apr 2024 11:52:18 -0700
## Body:
##
## Subject: objective 1
## Author: Jason Ballinger
## Date: Tue, 2 Apr 2024 14:07:37 -0700
## Body:
##
## Subject: Initial commit
## Author: Jason Ballinger
## Date: Fri, 29 Mar 2024 10:39:40 -0700
## Body:
```