Graded Assignment

Data Visualization 2024-2025

s4581733

Task 1. Data explorations (45%)

Task 1a)

```
# Import data:
# Using read_csv2 assuming the file uses semicolon as separator, as hinted by the function name.
vanEmissions <- read_csv2("Data/VanEmissionsLong.csv")
```

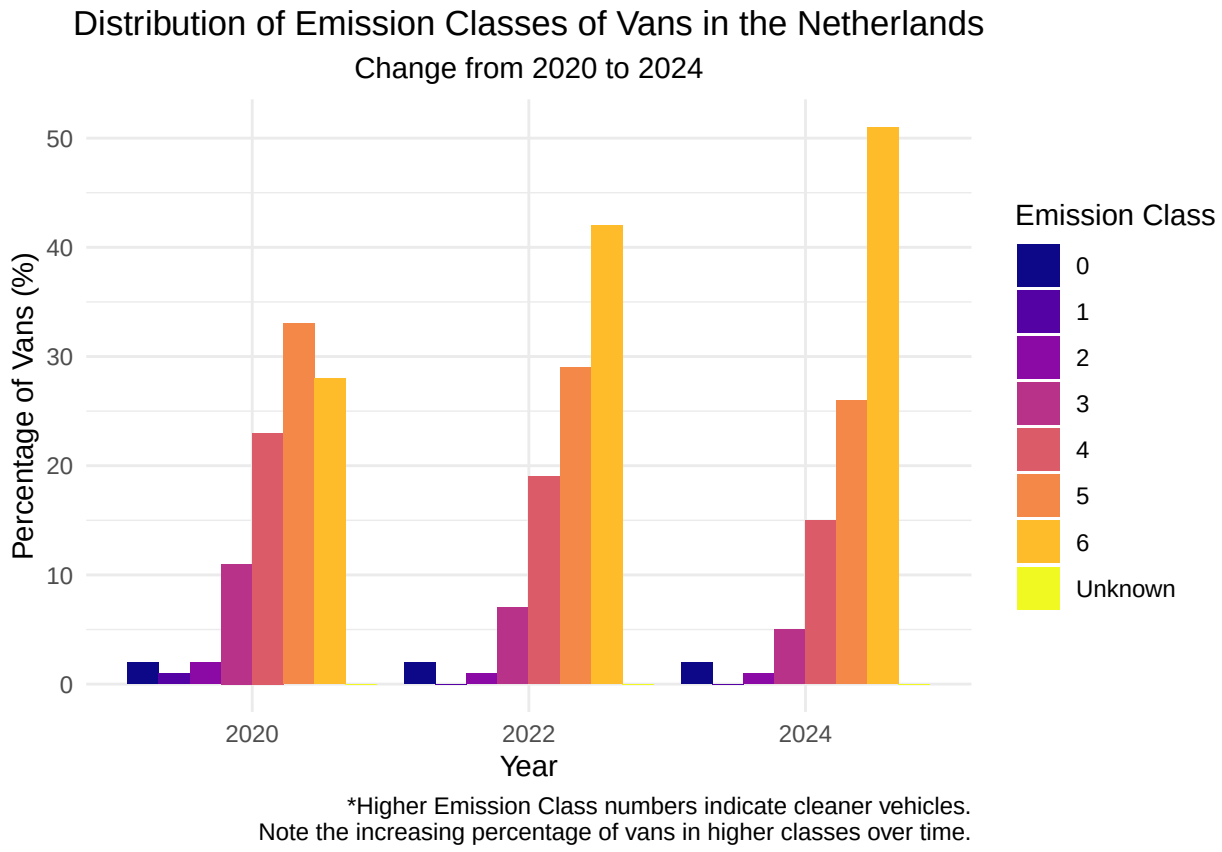
```
## i Using "','" as decimal and "'.'" as grouping mark. Use `read_delim()` for more control.
```

```
## Rows: 24 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (1): EmissionClass
## dbl (2): Year, Percentage
##
## 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.
```

```
# Convert relevant columns to factors for correct plotting
vanEmissions$Year <- factor(vanEmissions$Year)
# Ensure EmissionClass is treated as ordered factor if higher implies cleaner,
# or simply as factor for discrete colors. Let's treat it as factor for fill.
vanEmissions$EmissionClass <- factor(vanEmissions$EmissionClass,
                                     levels = sort(unique(vanEmissions$EmissionClass))) # Ensure classe

# Add your code here
# Check data structure
str(vanEmissions)
```

```
## spc_tbl_ [24 x 3] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Year         : Factor w/ 3 levels "2020","2022",..: 3 3 3 3 3 3 3 3 2 2 ...
##  $ EmissionClass: Factor w/ 8 levels "0","1","2","3",..: 8 1 2 3 4 5 6 7 8 1 ...
##  $ Percentage   : num [1:24] 0 2 0 1 5 15 26 51 0 2 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Year = col_double(),
##   ..   EmissionClass = col_character(),
##   ..   Percentage = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```r
# Add your code here
# Create the grouped bar chart
ggplot(vanEmissions, aes(x = Year, y = Percentage, fill = EmissionClass)) +
  geom_col(position = "dodge") + # Use position = "dodge" to group bars by year
  scale_fill_viridis_d(option = "plasma") + # Use a discrete color scale suitable for different classes
  labs(
    title = "Distribution of Emission Classes of Vans in the Netherlands",
    subtitle = "Change from 2020 to 2024",
    x = "Year",
    y = "Percentage of Vans (%)",
    fill = "Emission Class", # Legend title for fill color
    caption = "*Higher Emission Class numbers indicate cleaner vehicles.\nNote the increasing percentag
  ) +
  theme_minimal() + # Use a clean theme
  theme(
    plot.title = element_text(hjust = 0.5), # Center plot title
    plot.subtitle = element_text(hjust = 0.5) # Center plot subtitle
  )
```




```r
# Import data:
# Using read_csv2 assuming the file uses semicolon as separator.
speedTickets2023 <- read_csv2("Data/speedTickets_perMonthPerProvince.csv")
```

```
## i Using "','" as decimal and "'.'" as grouping mark. Use `read_delim()` for more control.
```

```
## Rows: 144 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (2): Province, geometry
## dbl (2): Month, Count
##
## 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.
```

```
# Convert Month to ordered factor or numeric if treating as time series.
# Since months are ordered, converting to factor might be better for discrete x-axis positions.
speedTickets2023$Month <- factor(speedTickets2023$Month, levels = 1:12,
                                 labels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep

str(speedTickets2023)
```

```
## spc_tbl_ [144 x 4] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Province: chr [1:144] "Drenthe" "Flevoland" "Friesland" "Gelderland" ...
##  $ Month   : Factor w/ 12 levels "Jan","Feb","Mar",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Count   : num [1:144] 2105 5458 790 22382 9286 ...
##  $ geometry: chr [1:144] "list(list(c(232285.471929562, 231251.917931354, 231261.743475229, 231568.28
##  - attr(*, "spec")=
##   .. cols(
##   ..   Province = col_character(),
##   ..   Month = col_double(),
##   ..   Count = col_double(),
##   ..   geometry = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

Task 1b)

```
# Import data:
# Using read_csv2 assuming the file uses semicolon as separator.
speedTickets2023 <- read_csv2("Data/speedTickets_perMonthPerProvince.csv")
```

```
## i Using "','" as decimal and "'.'" as grouping mark. Use `read_delim()` for more control.
```

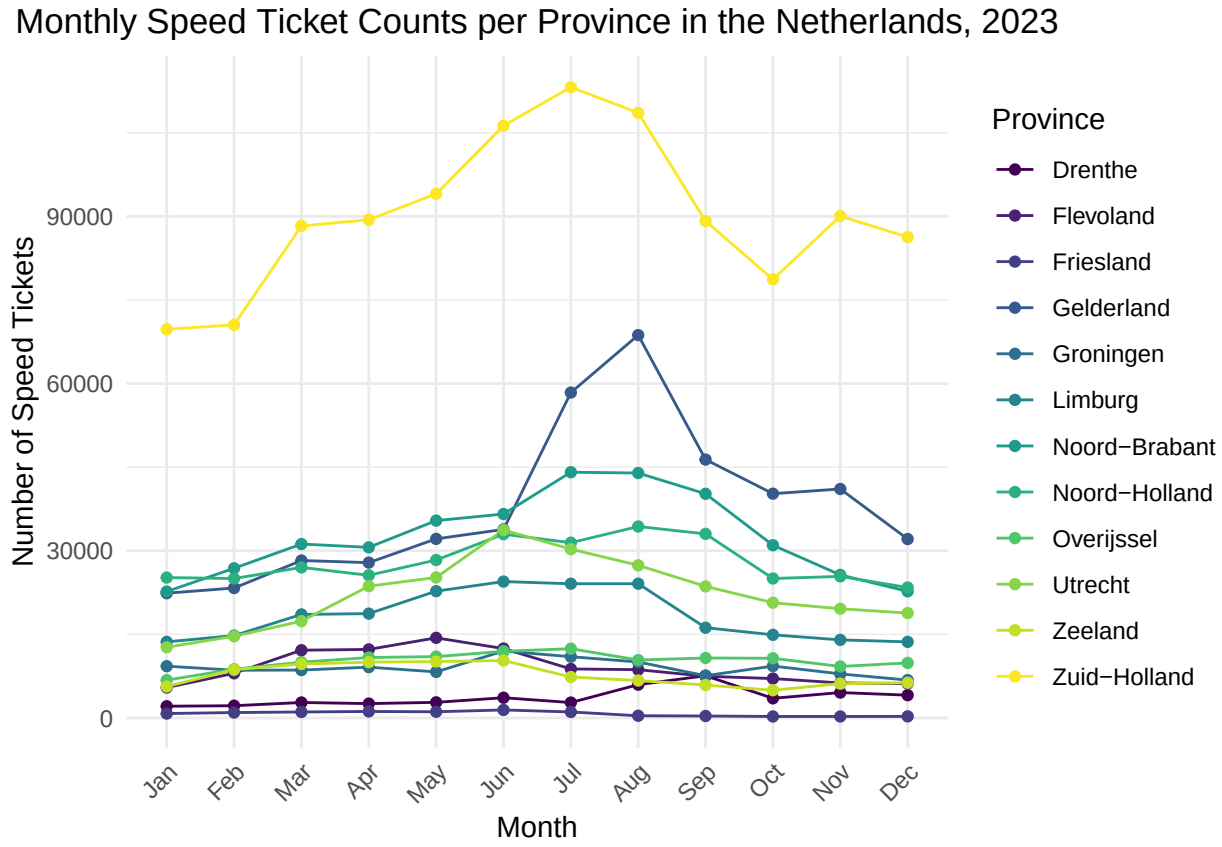
```
## Rows: 144 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (2): Province, geometry
## dbl (2): Month, Count
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# Convert Month to ordered factor or numeric if treating as time series.
# Since months are ordered, converting to factor might be better for discrete x-axis positions.
speedTickets2023$Month <- factor(speedTickets2023$Month, levels = 1:12,
```

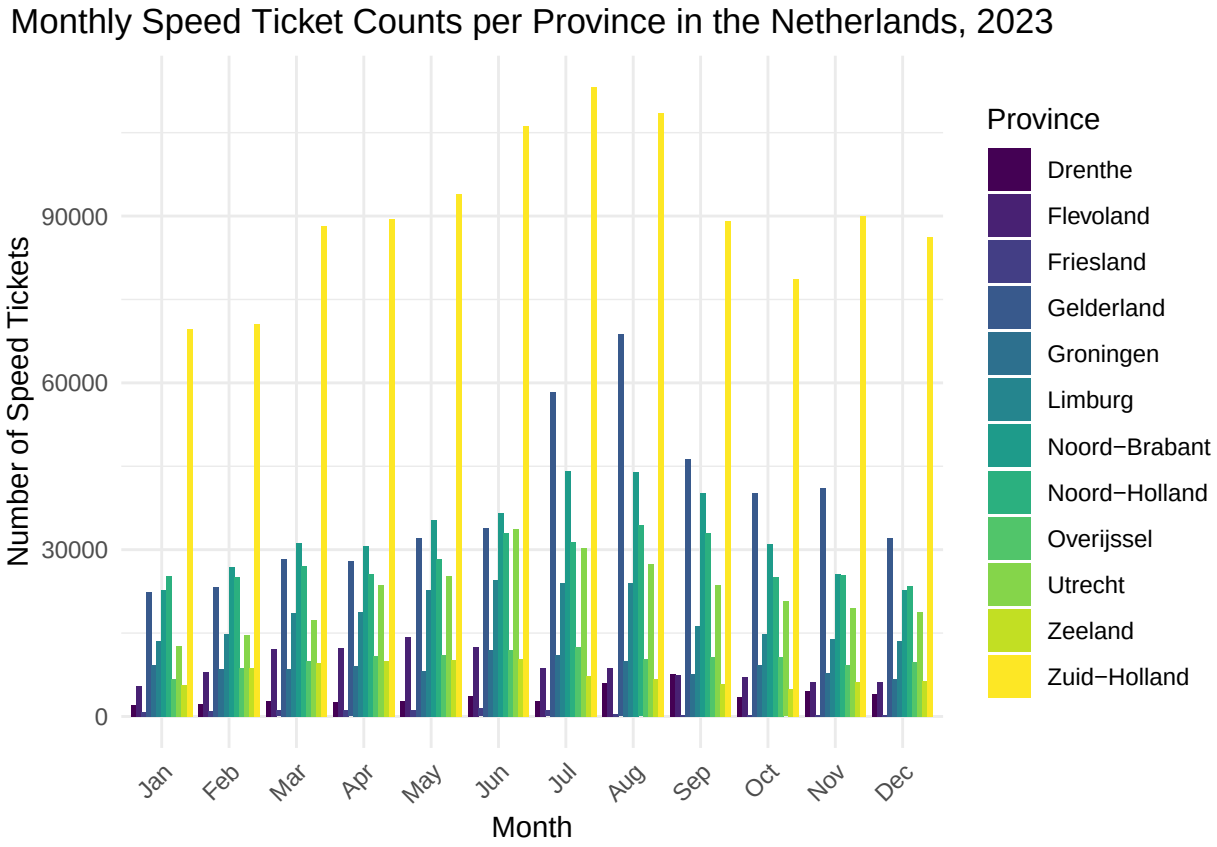
```
                                  labels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep

# Province should be a factor
speedTickets2023$Province <- factor(speedTickets2023$Province)

# geometry column is present but not needed for this specific plot
speedTickets2023 <- select(speedTickets2023, -geometry) # Remove geometry column if not needed for this

# Add your code here
str(speedTickets2023)
```

```
## tibble [144 x 3] (S3: tbl_df/tbl/data.frame)
##  $ Province: Factor w/ 12 levels "Drenthe","Flevoland",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Month   : Factor w/ 12 levels "Jan","Feb","Mar",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Count   : num [1:144] 2105 5458 790 22382 9286 ...
```

```
# Create the plot showing ticket counts over months for each province
# Using a line plot to show trends over time for each province
ggplot(speedTickets2023, aes(x = Month, y = Count, group = Province, color = Province)) +
  geom_line() + # Draw lines connecting monthly counts for each province
  geom_point() + # Add points for each data point
  scale_color_viridis_d() + # Use a discrete color scale for provinces
  labs(
    title = "Monthly Speed Ticket Counts per Province in the Netherlands, 2023",
    x = "Month",
    y = "Number of Speed Tickets",
    color = "Province" # Legend title for color
  ) +
  theme_minimal() +
   theme(
    plot.title = element_text(hjust = 0.5), # Center plot title
    axis.text.x = element_text(angle = 45, hjust = 1) # Angle x-axis labels for readability
  )
```




```r
# Alternative using grouped bars to easily compare provinces within each month (as in initial template
# This also makes it easy to visually identify provinces with consistently higher bars
ggplot(speedTickets2023, aes(x = Month, y = Count, fill = Province)) +
  geom_col(position = "dodge") +
  scale_fill_viridis_d() +
  labs(
    title = "Monthly Speed Ticket Counts per Province in the Netherlands, 2023",
    x = "Month",
    y = "Number of Speed Tickets",
    fill = "Province"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1)
  )
```




```
# Choose one of the plots above. The line plot shows trends better,
# the grouped bar chart makes direct monthly comparisons between provinces easier.
# The prompt asks how the number changes over time *for each province* (line plot is good)
# and which provinces are *generally* highest (both can show this, grouped bar might be clearer visuall
# Let's keep the grouped bar chart as it directly shows counts side-by-side per month.
```

Task 1. c)

```
# Load data as a data frame, set variables to numeric
# Using read.csv2 assuming semicolon separator. If comma, use read.csv
# The prompt uses read.csv2 but variable descriptions use '.', assume read.csv2 is correct.
# Convert columns explicitly to numeric, coercing errors to NA.
marvel <- read.csv2("Data/Marvel.csv")
marvel$AudienceScore <- as.numeric(as.character(marvel$AudienceScore)) # Convert factor to character fi
marvel$CriticsScore <- as.numeric(as.character(marvel$CriticsScore))   # Convert factor to character fi
# WorldGross seems to already be numeric in the provided data glimpse in the template
# If not, convert: marvel$WorldGross <- as.numeric(as.character(marvel$WorldGross))

# Convert Category to factor
marvel$Category <- factor(marvel$Category)

# Add your code here
str(marvel)
```

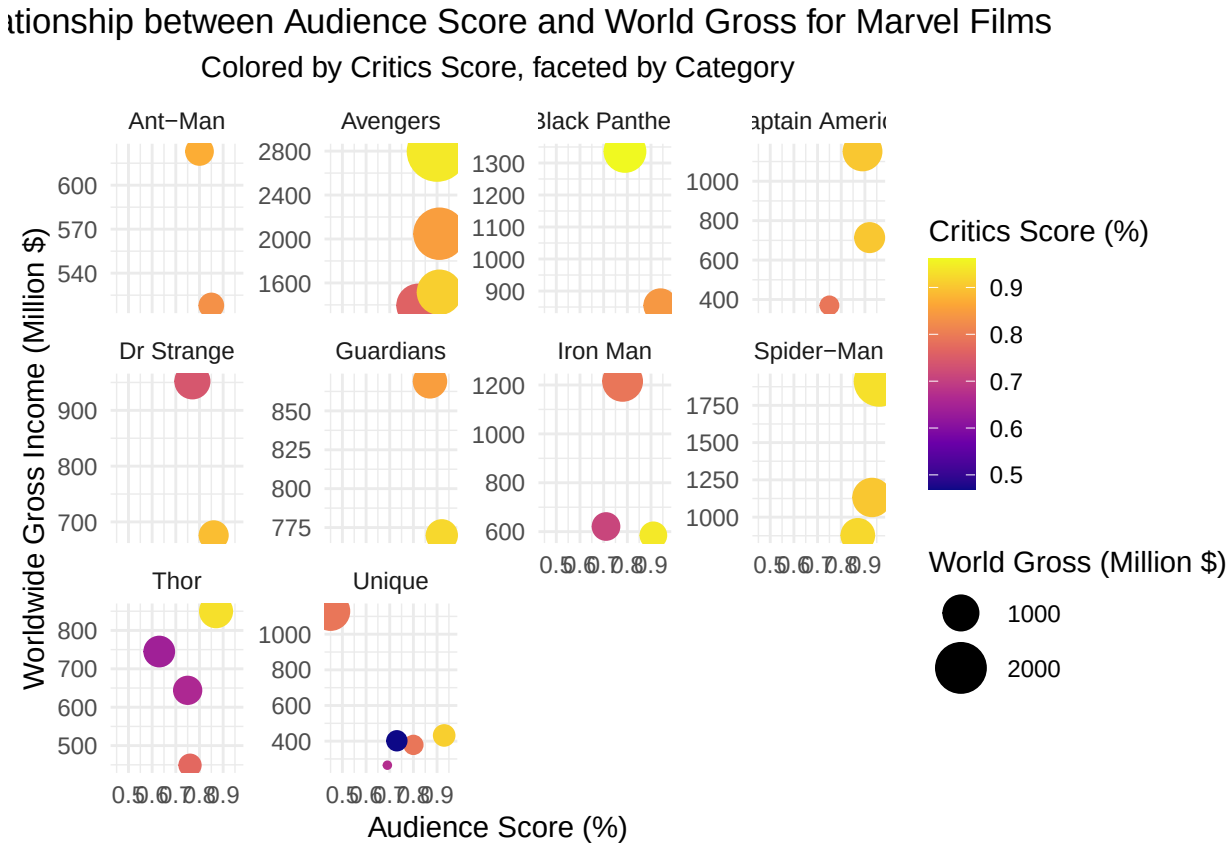
```
## 'data.frame':    30 obs. of  18 variables:
##  $ Film                         : chr  "Ant-Man  " "Ant-Man & The Wasp" "Avengers: Age of Ultron" "A
##  $ Category                     : Factor w/ 10 levels "Ant-Man","Avengers",..: 1 1 2 2 2 3 3 10 4 4
##  $ WorldGross                   : int  518 623 1395 2797 2048 1336 855 379 370 1151 ...
##  $ PercentageBudgetRecovered    : chr  "398.46" "479.23" "382.19" "699.25" ...
##  $ CriticsScore                 : num  0.83 0.87 0.76 0.94 0.85 0.96 0.84 0.79 0.79 0.9 ...
##  $ AudienceScore                : num  0.85 0.8 0.82 0.9 0.91 0.79 0.94 0.8 0.75 0.89 ...
##  $ AudienceVSCritics            : chr  "-0.02" "0.07" "-0.06" "0.04" ...
##  $ Budget                       : chr  "130" "130" "365" "400" ...
##  $ DomesticGross                : int  180 216 459 858 678 700 453 183 176 408 ...
##  $ InternationalGross           : int  338 406 936 1939 1369 636 401 196 193 743 ...
##  $ OpeningWeekend               : chr  "57" "75.8" "191" "357" ...
##  $ SecondWeekend                : chr  "24" "29" "77" "147" ...
##  $ FirstVSSecWeekend            : chr  "-0.58" "-0.62" "-0.6" "-0.59" ...
##  $ PercGrossOpeningWeekend      : chr  "31.8" "35" "41.7" "41.6" ...
##  $ PercGrossDomestic            : chr  "0.35" "0.35" "0.33" "0.31" ...
##  $ PercGrossInternational       : chr  "0.65" "0.65" "0.67" "0.69" ...
##  $ PercBudgetOpeningWeekendudget: chr  "0.44" "0.58" "0.52" "0.89" ...
##  $ Year                         : int  2015 2018 2015 2019 2018 2018 2022 2021 2011 2016 ...
```

```r
summary(marvel) # Check for NAs introduced by conversion
```

```
##      Film                      Category   WorldGross
##  Length:30          Unique         :5   Min.   : 265.0
##  Class :character   Avengers       :4   1st Qu.: 594.0
##  Mode  :character   Thor           :4   Median : 810.0
##                     Captain America:3   Mean   : 940.9
##                     Iron Man       :3   3rd Qu.:1146.2
##                     Spider-Man     :3   Max.   :2797.0
##                     (Other)        :8
##  PercentageBudgetRecovered  CriticsScore    AudienceScore    AudienceVSCritics
##  Length:30                  Min.   :0.4700   Min.   :0.4500   Length:30
##  Class :character           1st Qu.:0.7750   1st Qu.:0.7625   Class :character
##  Mode  :character           Median :0.8500   Median :0.8550   Mode  :character
##                             Mean   :0.8257   Mean   :0.8223
##                             3rd Qu.:0.9100   3rd Qu.:0.9100
##                             Max.   :0.9600   Max.   :0.9600
##
##     Budget          DomesticGross   InternationalGross OpeningWeekend
##  Length:30          Min.   :134.0   Min.   : 130.0     Length:30
##  Class :character   1st Qu.:218.0   1st Qu.: 315.5     Class :character
##  Mode  :character   Median :333.5   Median : 448.5     Mode  :character
##                     Mean   :370.9   Mean   : 569.4
##                     3rd Qu.:422.2   3rd Qu.: 731.2
##                     Max.   :858.0   Max.   :1939.0
##
##  SecondWeekend      FirstVSSecWeekend  PercGrossOpeningWeekend
##  Length:30          Length:30          Length:30
##  Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character
##
##
##
##
```

```
##  PercGrossDomestic  PercGrossInternational PercBudgetOpeningWeekendudget
##  Length:30          Length:30              Length:30
##  Class :character   Class :character       Class :character
##  Mode  :character   Mode  :character       Mode  :character
##
##
##
##
##       Year
##  Min.   :2008
##  1st Qu.:2013
##  Median :2017
##  Mean   :2016
##  3rd Qu.:2019
##  Max.   :2022
##
```

```r
# Create the scatter plot
ggplot(marvel, aes(x = AudienceScore, y = WorldGross)) +
  geom_point(aes(color = CriticsScore, size = WorldGross)) + # Map CriticsScore to color, WorldGross to
  scale_color_viridis_c(option = "plasma", name = "Critics Score (%)") + # Color scale for Critics Scor
  scale_size_continuous(name = "World Gross (Million $)", range = c(1, 10)) + # Size scale for World Gr
  facet_wrap(~ Category, scales = "free_y") + # Create separate plots for each category, with free y-ax
  labs(
    title = "Relationship between Audience Score and World Gross for Marvel Films",
    subtitle = "Colored by Critics Score, faceted by Category",
    x = "Audience Score (%)",
    y = "Worldwide Gross Income (Million $)"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    plot.subtitle = element_text(hjust = 0.5)
  )
```



Task 2. Remake the plot (50%)

```
# Import data:
# Using st_read for shapefiles
potentialsolarenergyUSA <- st_read("Data/potentialsolarenergyUSA_shp/potentialsolarenergyUSA.shp")
```

```
## Reading layer 'potentialsolarenergyUSA' from data source
##   '/Users/adala/Downloads/s4581733/Data/potentialsolarenergyUSA_shp/potentialsolarenergyUSA.shp'
##   using driver 'ESRI Shapefile'
## Simple feature collection with 51 features and 2 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -179.1435 ymin: 18.90612 xmax: 179.7809 ymax: 71.4125
## Geodetic CRS:  WGS 84
```

```
# Check data structure
str(potentialsolarenergyUSA)
```

```
## Classes 'sf' and 'data.frame':   51 obs. of  3 variables:
##  $ State   : chr  "Alabama" "Alaska" "Arizona" "Arkansas" ...
##  $ SlrPtnt : num  3743 8283 11989 5015 9102 ...
##  $ geometry:sfc_MULTIPOLYGON of length 51; first list element: List of 3
```

```
##   ..$ :List of 1
##   .. ..$ : num [1:406, 1:2] -87.4 -87.4 -87.4 -87.4 -87.4 ...
##   ..$ :List of 1
##   .. ..$ : num [1:15, 1:2] -87.5 -87.5 -87.4 -87.4 -87.5 ...
##   ..$ :List of 1
##   .. ..$ : num [1:13, 1:2] -88.1 -88.1 -88.1 -88.3 -88.3 ...
##   ..- attr(*, "class")= chr [1:3] "XY" "MULTIPOLYGON" "sfg"
##  - attr(*, "sf_column")= chr "geometry"
##  - attr(*, "agr")= Factor w/ 3 levels "constant","aggregate",..: NA NA
##   ..- attr(*, "names")= chr [1:2] "State" "SlrPtnt"
```

```
summary(potentialsolarenergyUSA)
```

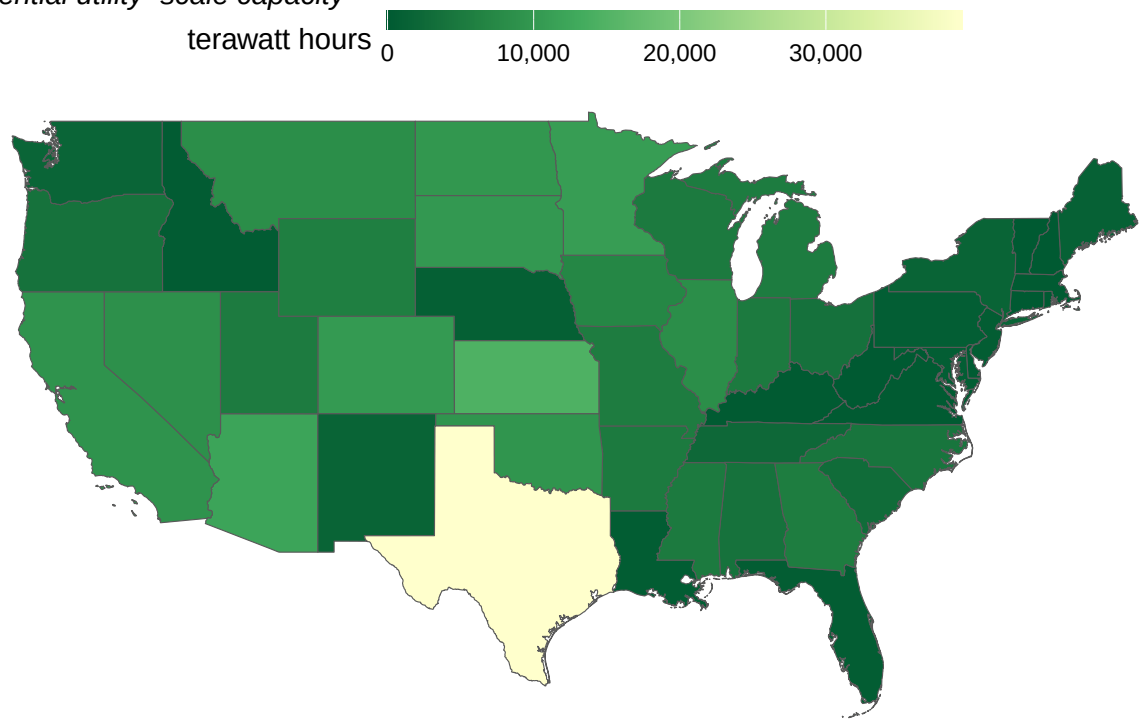
```
##     State              SlrPtnt                  geometry
##  Length:51          Min.   :    0.0   MULTIPOLYGON :51
##  Class :character   1st Qu.:  450.5   epsg:4326    : 0
##  Mode  :character   Median : 3743.0   +proj=long...: 0
##                     Mean   : 4791.2
##                     3rd Qu.: 7608.0
##                     Max.   :39288.0
```

```
# Add your code here

# Filter out Alaska ("Alaska") and Hawaii ("Hawaii")
contiguous_usa <- potentialsolarenergyUSA %>%
  filter(!State %in% c("Alaska", "Hawaii"))

# Calculate the total solar potential for the contiguous states (for the caption)
total_slr_ptnt <- sum(contiguous_usa$SlrPtnt, na.rm = TRUE)

# Create the choropleth map
ggplot(data = contiguous_usa) + # Use the filtered data frame
  geom_sf(aes(fill = SlrPtnt)) + # Map SlrPtnt to fill color, geom_sf handles the geometry
  scale_fill_distiller(
    palette = "YlGn", # Use the 'Greens' sequential palette
    direction = -1,      # Use direction = 1 for darker colors representing higher values (standard for
    trans = "identity",      # Keep log transformation
    breaks = c(0, 10000, 20000, 30000, 40000), # Keep breaks
    labels = scales::comma, # Keep labels format
    name = "terawatt hours", # Keep legend title
    guide = guide_colorbar( # Use guide_colorbar to control legend appearance
      barwidth = 15,      # Set the width of the color bar (adjust value as needed)
      barheight = 0.5       # Set the height of the color bar (adjust value as needed)
    )
  ) +
  theme_void() + # Remove map grid and axes
  labs(
    title = "The USA is Ripe for Solar Power",
    subtitle = "Potential utility-scale capacity",
    caption = paste("Total:", format(round(total_slr_ptnt), big.mark = ","), "terawatt hours") # Includ
  ) +
  theme(
    legend.position = "top", # Place legend at the top
```

```
  plot.title.position = "panel", # Center plot title
  plot.title = element_text(face = "bold"),
  plot.subtitle = element_text(face = "italic"), # Center plot subtitle
  plot.caption = element_text(hjust = 1) # Align caption to the right
)
```

The USA is Ripe for Solar Power

Potential utility−scale capacity



Total: 236,025 terawatt hours