# PSY 6422 Project

2024-12-07

## Contents



## 1 Project Statement

The primary inspiration of this project was to understand external factors that may contribute to depression. With research into available databases and existing data, the topic was further refined to include medical status and existing conditions. This resulted in the following question, "To what extend do different medical statuses, including availability and visitations, affect depression with respect to perceived physical wellbeing?"

## 2 Packages

A host of different packages were used throughout the process. This and much of the code was split across four different files, separated for organisational purposes, but is presented here in full.

## 3 Data Origin

The data originated from the CDC National Center for Health Statistics (2024). The data was split across four different .xpt files, relating to the respective questionnaires as described below.

| File Name | Basic Description |
|---|---|
| DPQ_L.xpt | Depression Screening |
| HIQ_L.xpt | Insurance Information |
| HUQ_L.xpt | Haspital Usage |

| File Name | Basic Description |
|---|---|
| MCQ_L.xpt | Diagnosed Conditions |

https://wwwn.cdc.gov/nchs/nhanes/search/datapage.aspx?Component=Questionnaire&Cycle

# 4 Data Parsing

Parsing the data done in the following function, it processes the .xpt files into dataframe objects. The function also utilises webscraping elements to display information about the specific variables and is specifically built to work with any CDC questionnaire, intended for extended repeatability.

```
# custom function for parsing data (takes cdc file format xpt and documentation from cdc website)
df_parser <- function(data_file, metadata) {
  # print file name
  print(str_sub(data_file, start = 6, end = -5))
  # load in file
  df_raw <- read_xpt(data_file)
  # clean all rows and columns that only contain NA
  df <- df_raw[rowSums(is.na(df_raw)) != ncol(df_raw)-1, colSums(is.na(df_raw))<nrow(df_raw)]
  # webscrape documentation as meta
  tryCatch(
    {
      meta <- read_html(metadata)
      # create dataframe with pertinent information
      info <- data.frame(variable=character(), question=character(), data_type=character())
      # loop to parse relevant descriptions for column codes
      for (i in colnames(df)) {
        # get column class
        col_class <- class(df[[i]])
        # get html code containing code description
        title_meta <- meta %>% html_elements(xpath=glue("//*[contains(@id, '{i}')]"))
        # corner case, manage stringe case error in if statement when title_meta is empty
        if (length(title_meta)==0) {
          # fix casing for i to collect metadata properly
          i <- paste(str_sub(i, start=1, end=-2), str_to_lower(str_sub(i, start=-1)), sep='')
          # attempt to collect title_meta again
          title_meta <- meta %>% html_elements(xpath=glue("//*[contains(@id, '{i}')]"))
        }
        # extract text from title_meta
        test_data <- html_text(title_meta)
        # extract final description for column code
        desc <- str_trim(str_split_1(test_data, '-')[2])
        # append column metadata to row
        info[nrow(info) + 1,] = c(i, desc, col_class)
      }
      # print metadata for dataframe
      print(info)
    },
    # state any errors that occur during webscraping
    error=function(e) {
      message('An Error Occurred')
      print(e)
```

```
    },
    # state any warnings that occur during webscraping
    warning=function(w) {
      message('A Warning Occurred')
      print(w)
    }
  )
  return(df)
}
```

### 4.0.1 Output From Data

```
[1] "DPQ_L"
   variable                               question data_type
1      SEQN             Respondent sequence number   numeric
2    DPQ010  Have little interest in doing things   numeric
3    DPQ020  Feeling down, depressed, or hopeless   numeric
4    DPQ030 Trouble sleeping or sleeping too much   numeric
5    DPQ040 Feeling tired or having little energy   numeric
6    DPQ050            Poor appetite or overeating   numeric
7    DPQ060             Feeling bad about yourself   numeric
8    DPQ070       Trouble concentrating on things   numeric
9    DPQ080 Moving or speaking slowly or too fast   numeric
10   DPQ090  Thought you would be better off dead   numeric
11   DPQ100 Difficulty these problems have caused   numeric
```

```
[1] "HIQ_L"
   variable                               question data_type
1      SEQN             Respondent sequence number   numeric
2    HIQ011            Covered by health insurance   numeric
3   HIQ032A           Covered by private insurance   numeric
4   HIQ032B                    Covered by Medicare   numeric
5   HIQ032C                        Covered by Medi   numeric
6   HIQ032D                    Covered by Medicaid   numeric
7   HIQ032E                        Covered by CHIP   numeric
8   HIQ032F       Covered by military health care   numeric
9   HIQ032H                       Covered by state   numeric
10  HIQ032I Covered by other government insurance   numeric
11   HIQ210  Time when no insurance in past year?   numeric
```

```
[1] "HUQ_L"
  variable                                question data_type
1     SEQN              Respondent sequence number   numeric
2   HUQ010                General health condition   numeric
3   HUQ030        Routine place to go for healthcare   numeric
4   HUQ042 Type place most often go for healthcare   numeric
5   HUQ055      Past 12 months had video conf w/Dr?   numeric
6   HUQ090 Seen mental health professional/past yr   numeric
```

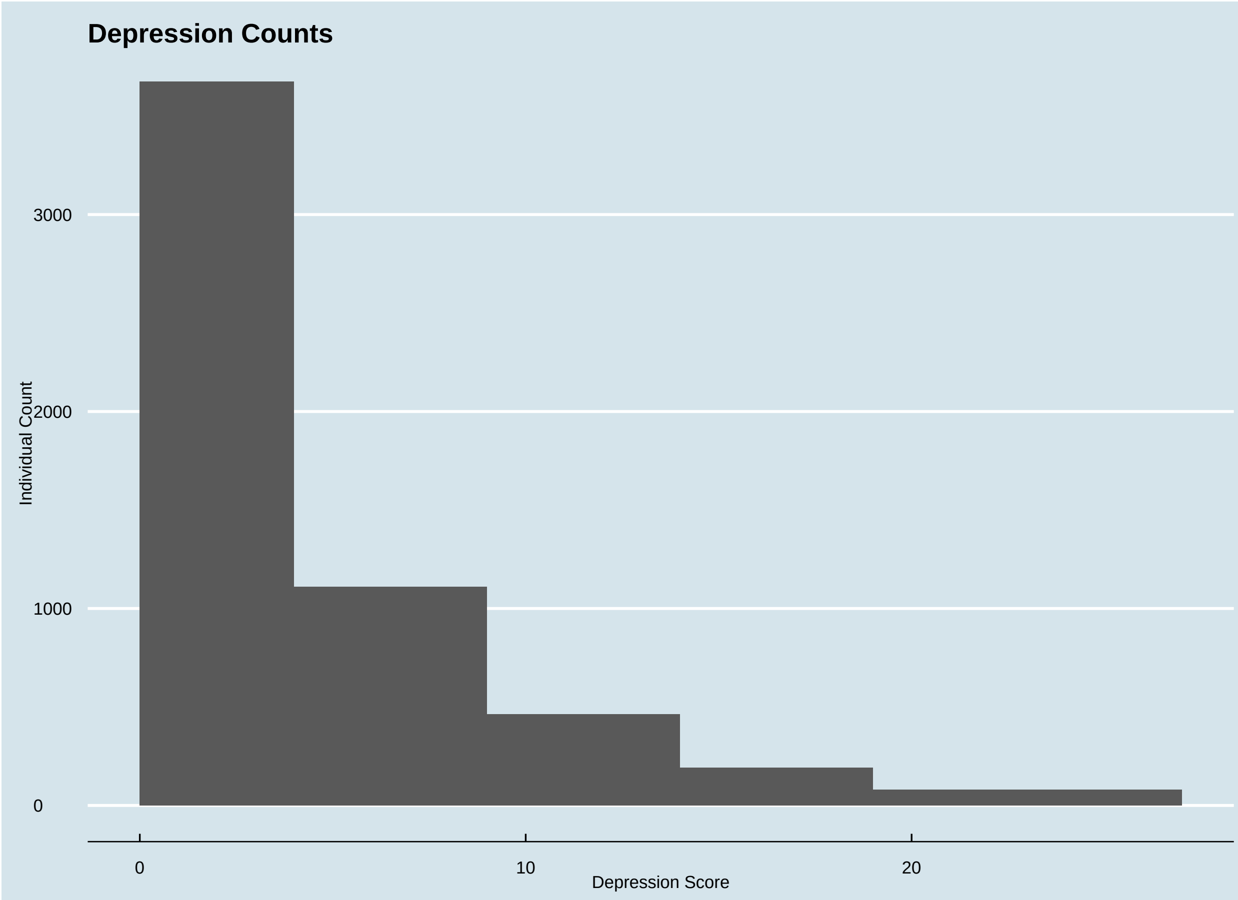
```
[1] "MCQ_L"
   variable                                 question data_type
1      SEQN               Respondent sequence number   numeric
2    MCQ010             Ever been told you have asthma   numeric
3    MCQ035                        Still have asthma   numeric
4    MCQ040             Had asthma attack in past year   numeric
```

```
5    MCQ050  Emergency care visit for asthma/past yr   numeric
6    AGQ030 Did SP have episode of hay fever/past yr   numeric
7    MCQ053   Taking treatment for anemia/past 3 mos   numeric
8    MCQ149           Menstrual periods started yet?   numeric
9   MCQ160a       Doctor ever said you had arthritis   numeric
10   MCQ195          Which type of arthritis was it?   numeric
11  MCQ160b   Ever told had congestive heart failure   numeric
12  MCQ160c Ever told you had coronary heart disease   numeric
13  MCQ160d Ever told you had angina/angina pectoris   numeric
14  MCQ160e           Ever told you had heart attack   numeric
15  MCQ160f               Ever told you had a stroke   numeric
16  MCQ160m        Ever told you had thyroid problem   numeric
17  MCQ170m        Do you still have thyroid problem   numeric
18  MCQ160p   Ever told you had COPD, emphysema, ChB   numeric
19  MCQ160l    Ever told you had any liver condition   numeric
20  MCQ170l      Do you still have a liver condition   numeric
21   MCQ500    Ever told you had any liver condition   numeric
22  MCQ510a              Liver condition: Fatty liver   numeric
23  MCQ510b           Liver condition: Liver fibrosis   numeric
24  MCQ510c         Liver condition: Liver cirrhosis   numeric
25  MCQ510d         Liver condition: Viral hepatitis   numeric
 [ reached 'max' / getOption("max.print") -- omitted 10 rows ]
```

# 5 Data Cleaning and Preperation for Analysis

## 5.1 Initial Cleaning and Inferences

After loading in the data, the values from each of the tables were standardised and combined. This would include recoding numerical data to the appropriate categorical responses (i.e. Yes/No responses, Likert Scale Higest to Lowest), summing the scores from the depression questionnaire, and merging the dataframes based on their user id. The depressions summed scores were also recoded to the appropriate output according to Kroenke et. al. (2001) and displayed as a simple histogram for further analysis.




### 5.2 Statistical Significance

As displayed in the prior section, the depression scores are exponentially distributed. Since the dependent variable is non-parametric, significance testing was conducted using the appropriate tests. The columns and values were run through the following function which is automated to calculate the significance based on the dependent variable type and independent variable type. This would result in either a chi squared test, mann witney test, or kruskal wallis test, depending on the variable type.

```r
# Function that automatically tests statistical significance based on data type
df_significance_testing <- function(df, dv, index) {
  print('Finding significant columns...')
  # Create empty list to return significant variables
  affective_columns <- c()
  if (class(df[[dv]])=='factor') {  #check for categorical dependent variable
    print('Dependent variable is categorical')
    for (i in colnames(df)) {
      # skips unnecessary columns
      if (i==index|i==dv|i=='dv') {
        next
      }
      print(glue('Running test with {i} column...'))
      # isolate dependent variable
      df_i <- df[c(dv, i)]
      colnames(df_i)[1:2] <- c('dv', 'iv')
      # filter missing data
```

```
      df_i <- df_i %>% filter(iv!='Missing')
      # run chi squared test on the data
      results <- df_run_chi_squared(df_i)
      # error filtering
      if (class(results)!="htest") {
        next
      # add columns as significant and skip insigificant columns based on p value
      } else if (results$p.value<0.05) {
        print(results)
        affective_columns <- c(affective_columns, i)
      } else {
        next
      }
    }
  } else if (class(df[[dv]])=='integer'|class(df[[dv]])=='numeric') { # check if dependent variable is 
    print('Dependent variable is numerical')
    for (i in colnames(df)) {
      # skip unnecessary columns
      if (i==index|i==dv|i=='dv') {
        next
      }
      print(glue('Running test with {i} column...'))
      if (identical(sort(unique(df[[i]])), c("Missing", "No", "Yes"))) {
        # isolate dependent variable
        df_i <- df[c(dv, i)]
        colnames(df_i)[1:2] <- c('dv', 'iv')
        # filter missing data
        df_i <- df_i %>% filter(iv!='Missing')
        # run mann witney test on the data
        results <- df_run_mann_witney(df_i)
        # error filtering
        if (class(results)!="htest") {
          print(class(results))
          next
        # add columns as significant and skip insigificant columns based on p value
        } else if (results$p.value<0.05) {
          affective_columns <- c(affective_columns, i)
        }
      } else {
        # isolate dependent variable
        df_i <- df[c(dv, i)]
        colnames(df_i)[1:2] <- c('dv', 'iv')
        # filter missing data
        df_i <- df_i %>% filter(iv!='Missing')
        # run kruskal wallis test on the data
        results <- kruskal.test(df_i$iv, df_i$dv)
        # error filtering
        if (class(results)!="htest") {
          print(class(results))
          next
        # add columns as significant and skip insigificant columns based on p value
        } else if (results$p.value<0.05) {
          print(results)
```

```
          affective_columns <- c(affective_columns, i)
        }
      }
    }
  } else {
    # Error handling for unsupported data types
    print(glue('unsupported dependent variable type: {class(df[[dv]]}'))
  }
  # return significant columns
  return(affective_columns)
}
```

The following variables were found to be significant as a result:

```
 [1] "HIQ210"     "gen_health" "HUQ090"     "MCQ160A"    "MCQ160B"
 [6] "MCQ160D"    "MCQ160F"    "MCQ160M"    "MCQ160P"    "MCQ160L"
[11] "MCQ550"     "OSQ230"
```

### 5.3 Ordinal Logistic Regression

Using the previously collected variables, an ordinal logistic regression model was selected to represent the data. The model was based on the UCLA Statistical Consulting Group's instructions (n.d.), to ensure consistent multivariate analysis on the categorical data. Variables were further tested against the model for significance, calculated based on the confidence intervals. Following this stage, the data can be graphed.

```
                            OR      2.5 %    97.5 %
HIQ210Yes            1.4706558  0.9665537  2.202318
gen_healthfair       8.2235296  5.7454967 12.003211
gen_healthgood       3.5358519  2.5307076  5.051517
gen_healthpoor      20.2958995 12.7061316 32.809755
gen_healthvery_good  1.4771788  1.0384697  2.142126
HUQ090Yes            3.9703690  3.2624032  4.829800
MCQ160AYes           1.3404114  1.1356040  1.582124
MCQ160BYes           1.0018018  0.7213839  1.381096
MCQ160DYes           0.8791560  0.5785356  1.318496
MCQ160FYes           1.2695492  0.9427639  1.698645
MCQ160MYes           1.3749366  1.1338553  1.663364
MCQ160PYes           1.2562921  0.9857002  1.596118
MCQ160LYes           1.0866502  0.8242280  1.424059
MCQ550Yes            1.2150954  0.9792827  1.502608
OSQ230Yes            0.9678383  0.8194050  1.142081
```

## 6 Visualisation and Analysis

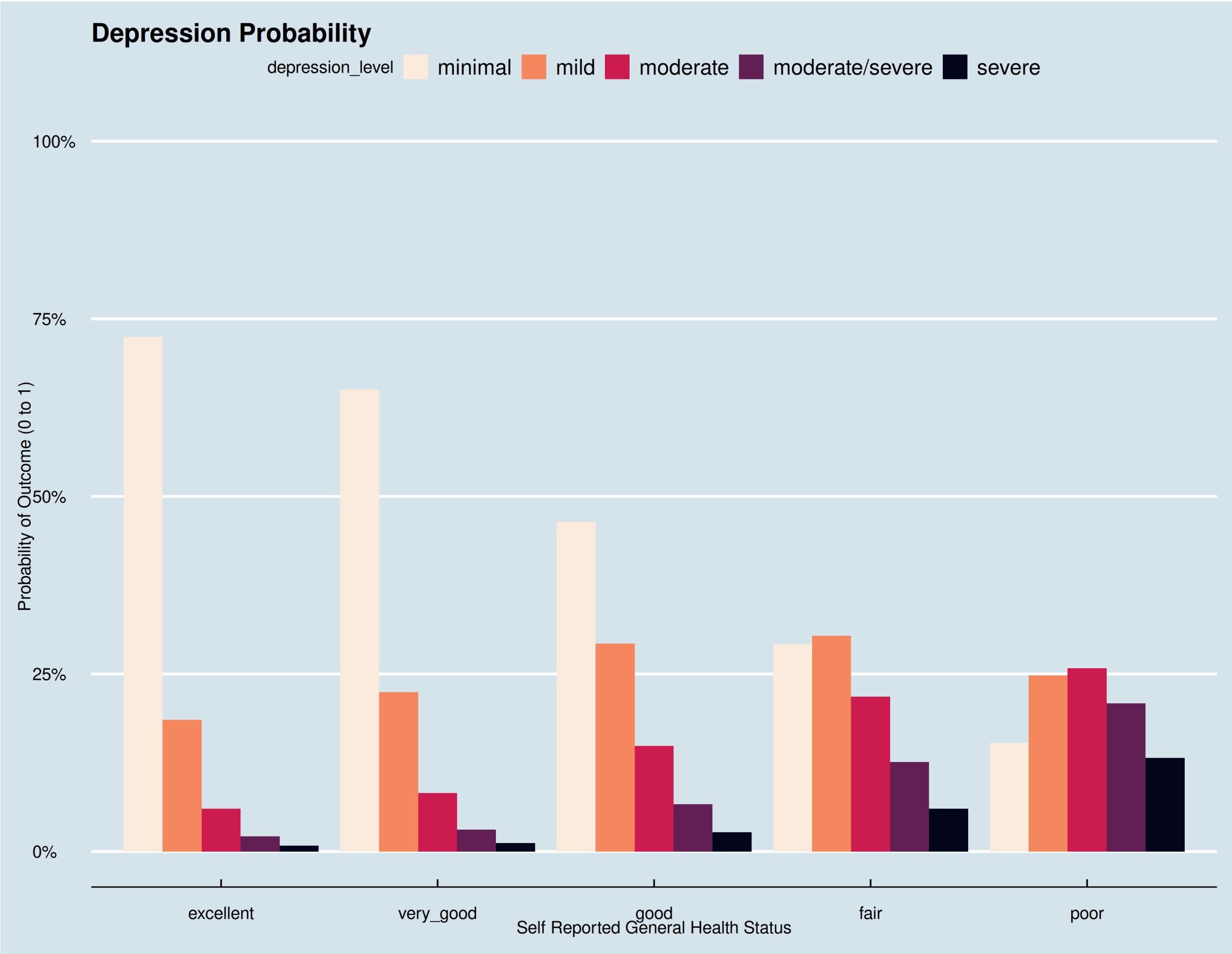
### 6.1 Creating the Visualisation

Graphin was conducted using the following function. It is is designed to output a bar graph with the x variable representing the general perceived health, the y variable representing the probability as a percentage, and multiple bars per x category to represent the depression score category outcome. The function can be used primarily in a shiny live application, which allows for variable filtering, but can still output a static application with hover capability.

```
graphing_scores <- function(df=df, name='base') {
  output_plot <- ggplot(df, mapping=aes(
    # set general health to the categorical x value
```

```r
    x = factor(gen_health, levels = c(
      'Excellent'='excellent',
      'Very Good'='very_good',
      'Good'='good',
      'Fair'='fair',
      'Pool'='poor'
    )),
    # set the probability to the y variable
    y = Probability,
    # set fill to be the depression level, making a separate bar for each
    fill = depression_level,
    # create tooltip to display the probability percent when hovered over
    tooltip = glue('Probability: {round(Probability, 4)*100}%'),
    # assign interactive element to probability variable
    data_id = Probability
    # create the interactive bar chart
  )) + geom_bar_interactive(position = 'dodge', stat = 'identity') + labs(
    # labeling
    title = 'Depression Probability',
    x = 'Self Reported General Health Status',
    y = 'Probability of Outcome (0 to 1)',
    # set hover to focus on mouse cursor
    hover_nearest = TRUE,
    aes(name='Depression Categorical depression_level')
    # scale the probabilities as percentages
  ) + scale_y_continuous(labels = scales::percent, limits = c(0,1)) +
    # add theaming
    theme_economist() +
    scale_fill_viridis(discrete = TRUE, direction = -1, option = "rocket")
  # export interactive plot element
  interactive_plot <- ggiraph(ggobj=output_plot, width_svg = 11, height_svg = 8.5)
  if (name != '') {
    # export graph as html files
    htmltools::save_html(interactive_plot, glue('figs/{name}.html'))
  }
  return(interactive_plot)
}
```

## 6.2 Final Visualisation




The graph includes interactive filtering options using the following block of code.

```
# add labels for the filter option
filter_options_labeled <- c(
  'Uninsured Past Year'='HIQ210',
  'Seen a Mental Health Professional Past Year'='HUQ090',
  'Arthritis'='MCQ160A',
  'Congestive Heart Failure'='MCQ160B',
  'Angina'='MCQ160D',
  'Stroke'='MCQ160F',
  'Thyroid Problems'='MCQ160M',
  'COPD/Emphasema/ChB'='MCQ160P',
  'Liver Condition'='MCQ160L',
  'Gallstones'='MCQ550',
  'Metal in Body'='OSQ230'
)

# function to initialise graphing and implementation of shiny elements
run_app <- function(df, filter_options, filter_options_labeled) {
  # create ui element for shiny chart
  ui <- fluidPage(
    # add theming to page
```

```r
    theme = shinytheme("flatly"),
    sidebarLayout(
      sidebarPanel(
        # create check box filter option
        checkboxGroupInput("cols", "Select Conditions:",
                           choices = filter_options_labeled, selected = filter_options
        # create table element with relevant data
        ), tableOutput(outputId = 'table'), width = 2),
      mainPanel(
        # create interactive graph element
        girafeOutput(outputId = "interactivePlot", width = '100%', height = NULL)
      )
    ),
  )

  # add server element to shiny plot for filtering
  server <- function(input, output) {
    # create a filterig function
    filtered_data <- reactive({
      # create a new dataframe for filtering the output
      graphing_df <- df
      # loop through the filter options
      for (i in filter_options) {
        # assign checked variables to "yes" result and unchecked to "no" result
        if (i == 'gen_health') {
          next
        } else if (i %in% input$cols){
          graphing_df <- graphing_df[graphing_df[[i]]=='Yes',]
        } else {
          graphing_df <- graphing_df[graphing_df[[i]]=='No',]
        }
      }
      # pivot the graph to group the appropriate format
      graphing_df %>% group_by(gen_health, depression_level) %>% summarise(Probability=mean(Probability)
    })
    # assign the interactive plot to the appropriate css tag
    output$interactivePlot <- renderGirafe({
      graphing_df <- data.frame(filtered_data())
      graphing_scores(graphing_df, name='')
    })
    # assign the table to the appropriate css tag
    output$table <- renderTable({
      table_df <- filtered_data()
      table_df$Probability <- scales::percent(table_df$Probability, accuracy = 0.01)
      table_df
    })
  }

  # run the shiny app as a local webpage
  shinyApp(ui = ui, server = server)
}
```

## 6.3 Analysis

According to the data, the primary variable correlated to a high depression score is having visited a mental health professional within the last year. All other variables, when this one is removed, is not high enough to surpass any other rank. This implies that having visited a mental health provider within a year is a greater indication of depression than any chronic condition or insurance status. However, lower general perceived health also lead to higher depression score rates. In all cases excellent general perceived health had minimal depression as the majority, while poor general percieved health ahd varying results including some where severe depression was the highest.

## 6.4 Limitations

There are some issues related to the data. Due to these questionnaire studies having the majority of questions unanswered and the necessities of the ordinal logistic regression model for complete data, almost half the data entries were removed. This may have significantly impacted the outcome of the study, decreasing the power and removing influential data points. This can be combated in the future by recording empty data points, however there are other concerns with that approach with regards to bias and precision.

## 6.5 Further Research

Potential research can be conducted by adding in more dates to the algorithm. Having dates be a factor, especially with COVID data being represented, can show a before and after to these trends. There are also other questionnaires, such as dietary or demographic data, that could be analysed within this context.


Arnold, J. B. (2024). *Ggthemes: Extra themes, scales and geoms for ggplot2*. https://jrnold.github.io/ggthemes/

Barnier, J. (2022). *Rmdformats: HTML output formats and templates for rmarkdown documents*. https://github.com/juba/rmdformats

Chang, W. (2021). *Shinythemes: Themes for shiny*. https://rstudio.github.io/shinythemes/

Chang, W., Cheng, J., Allaire, J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., & Borges, B. (2024). *Shiny: Web application framework for r*. https://shiny.posit.co/

Firke, S. (2023). *Janitor: Simple tools for examining and cleaning dirty data*. https://github.com/sfirke/janitor

Garnier, S. (2024). *Viridis: Colorblind-friendly color maps for r*. https://sjmgarnier.github.io/viridis/

Garnier, Simon, Ross, Noam, Rudis, Robert, Camargo, Pedro, A., Sciaini, Marco, Scherer, & Cédric. (2024). *viridis(Lite) - colorblind-friendly color maps for r*. https://doi.org/10.5281/zenodo.4679423

Gohel, D., & Skintzos, P. (2024). *Ggiraph: Make ggplot2 graphics interactive*. https://davidgohel.github.io/ggiraph/

Group, U. S. C. (n.d.). *Ordinal logistic regression in r*. https://stats.oarc.ucla.edu/r/dae/ordinal-logistic-regression/

Harrell, F. E., Jr. (2024). *Hmisc: Harrell miscellaneous*. https://hbiostat.org/R/Hmisc/

Hester, J., & Bryan, J. (2024). *Glue: Interpreted string literals*. https://glue.tidyverse.org/

Kroenke, K., Spitzer, R. L., & Williams, J. B. W. (2001). The PHQ-9: Validity of a brief depression severity measure. *Journal of General Internal Medicine*, *16*(9), 606–613. https://doi.org/10.1046/j.1525-1497.2001.016009606.x

National Center for Health Statistics. (2024). *National health and nutrition examination survey questionnaire*. Center for Disease Control; Prevention. https://wwwn.cdc.gov/nchs/nhanes/search/datapage.aspx?Component=Questionnaire&Cycle

R Core Team. (2024a). *Foreign: Read data stored by minitab, s, SAS, SPSS, stata, systat, weka, dBase, ...* https://svn.r-project.org/R-packages/trunk/foreign/

R Core Team. (2024b). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Ripley, B., & Venables, B. (2024). *MASS: Support functions and datasets for venables and ripley's MASS*. http://www.stats.ox.ac.uk/pub/MASS4/

Venables, W. N., & Ripley, B. D. (2002). *Modern applied statistics with s* (Fourth). Springer. https://www.stats.ox.ac.uk/pub/MASS4/

Wickham, H. (2016). *ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org

Wickham, H. (2023). *Stringr: Simple, consistent wrappers for common string operations*. https://stringr.tidyverse.org

Wickham, H. (2024). *Rvest: Easily harvest (scrape) web pages*. https://rvest.tidyverse.org/

Wickham, H., Chang, W., Henry, L., Pedersen, T. L., Takahashi, K., Wilke, C., Woo, K., Yutani, H., Dunnington, D., & van den Brand, T. (2024). *ggplot2: Create elegant data visualisations using the grammar of graphics*. https://ggplot2.tidyverse.org

Wickham, H., Miller, E., & Smith, D. (2023). *Haven: Import and export SPSS, stata and SAS files*. https://haven.tidyverse.org

Xie, Y. (2014). Knitr: A comprehensive tool for reproducible research in R. In V. Stodden, F. Leisch, & R. D. Peng (Eds.), *Implementing reproducible computational research*. Chapman; Hall/CRC.

Xie, Y. (2015). *Dynamic documents with R and knitr* (2nd ed.). Chapman; Hall/CRC. https://yihui.org/knitr/

Xie, Y. (2024). *Knitr: A general-purpose package for dynamic report generation in r*. https://yihui.org/knitr/