Health Information National Trends Survey 6 Data Visualization

Evan Shields

December 6, 2023

Load in necessary libraries

```
library(ggplot2)
library(dplyr)
library(tidyr)
library(stringr)
library(purrr)
library(tibble)
library(tidyverse)
```

Load in data file

```
load("hints6_public.rda")
data <- public
```

Columns of Interest

```
age <- data$Age
gender <- data$BirthGender
married <- data$MaritalStatus
education <- data$Education
# race
household <- data$TotalHousehold
exercise <- data$TimesModerateExercise
cancerEverything <- data$EverythingCauseCancer
cancerHad <- data$EverHadCancer
sunburned <- data$SUNBURNEDACT_CAT
```

Data Cleaning: Mainly turning missing data into NAs

```
# paste... combines the strings into a logical vector with |
# separators sunburned column subsets the logical vector to
# determine which levels have the substrings which function
# determines indices of TRUE and sets those levels to NA
removeAge <- str_detect(levels(age), paste(c("Missing data",
    "Unreadable"), collapse = "|"))
age[which(removeAge[age])] <- NA

removeGen <- str_detect(levels(gender), paste(c("Missing data",
    "Multiple"), collapse = "|"))
```

```
gender[which(removeGen[gender])] <- NA

removeMar <- str_detect(levels(married), paste(c("Missing data",
    "Multiple"), collapse = "|"))
married[which(removeMar[married])] <- NA

removeEdu <- str_detect(levels(education), paste(c("Missing data"),
    collapse = "|"))
education[which(removeEdu[education])] <- NA

# fct_collapse collapses (temporarily puts all levels on
# the same level) the household column str_detect is used
# to identify if the substring "Missing data" is in any of
# the levels, and if so, that level will be turned into NA
household <- household %>%
    fct_collapse(`NA` = levels(household)[str_detect(levels(household),
        "Missing data")])

removeSun <- str_detect(levels(sunburned), paste(c("Missing data",
    "Question", "Inapplicable"), collapse = "|"))
sunburned[which(removeSun[sunburned])] <- NA

exercise <- exercise %>%
    fct_collapse(`NA` = levels(exercise)[str_detect(levels(exercise),
        "Missing data")])
exercise <- as.character(exercise)

# assign 'None' response to 0 times per week
exercise[exercise == "None"] <- 0

# turn word responses into numbers
exercise <- gsub(".*?(\\d+).*", "\\1", exercise)

removeEve <- str_detect(levels(cancerEverything), paste(c("Missing data",
    "Multiple"), collapse = "|"))
cancerEverything[which(removeEve[cancerEverything])] <- NA

removeHad <- str_detect(levels(cancerHad), paste(c("Missing data"),
    collapse = "|"))
cancerHad[which(removeHad[cancerHad])] <- NA
```

Turning the 9 Selected/Not Race columns into one column with all necessary info

```
# select desired columns, pivot_longer puts all data from
# these 9 columns into 1 column
race <- data %>%
    select("White", "Black", "AmerInd", "AsInd", "Chinese", "Filipino",
        "Vietnamese", "OthAsian", "OthPacIsl") %>%
    pivot_longer(cols = everything(), names_to = "Race", values_to = "Selected")

# replaces entries with substring 'Missing data' with NA
race <- race %>%
    mutate(Selected = if_else(str_detect(Selected, "Missing data"),
```

```
        "NA", Selected))

# looping through race selection row and adding to and
# extending 2d list for every selected race. Each index in
# the first dimension is one of the 6252 observations, and
# each index in the 2nd dimension is the number of races
# someone selected
i <- 1
vec <- vector("list", length = ceiling(nrow(race)/9))
while (i <= nrow(race)) {
    if (race$Selected[i] == "Selected") {
        index <- floor((i - 1)/9) + 1
        if (is.null(vec[[index]])) {
            vec[[index]] <- list(race$Race[i])
        } else {
            vec[[index]] <- list(vec[[index]], race$Race[i])
        }
    }
    i <- i + 1
}

# iterates through the 2d list and unlists each index
# (turns vec into a 1d list)
vec <- map(vec, unlist)

df <- tibble(list_column = vec)

# concatenates the contents of each sublist into a single
# column
df <- df %>%
    mutate(race = map_chr(list_column, toString)) %>%
    select(-list_column)

# replaces empty characters with NA
df <- df %>%
    mutate(race = ifelse(race == "", NA, race))
```

Construct Data frame with columns of interest

```
df$age <- age
df$gender <- gender
df$married <- married
df$education <- education
df$household <- household
df$sunburned <- sunburned
df$exercise <- exercise
df$cancerEverything <- cancerEverything
df$cancerHad <- cancerHad

# change class of numerical columns
df$age <- as.character(df$age)
df$age <- as.integer(df$age)
df$household <- as.character(df$household)
```

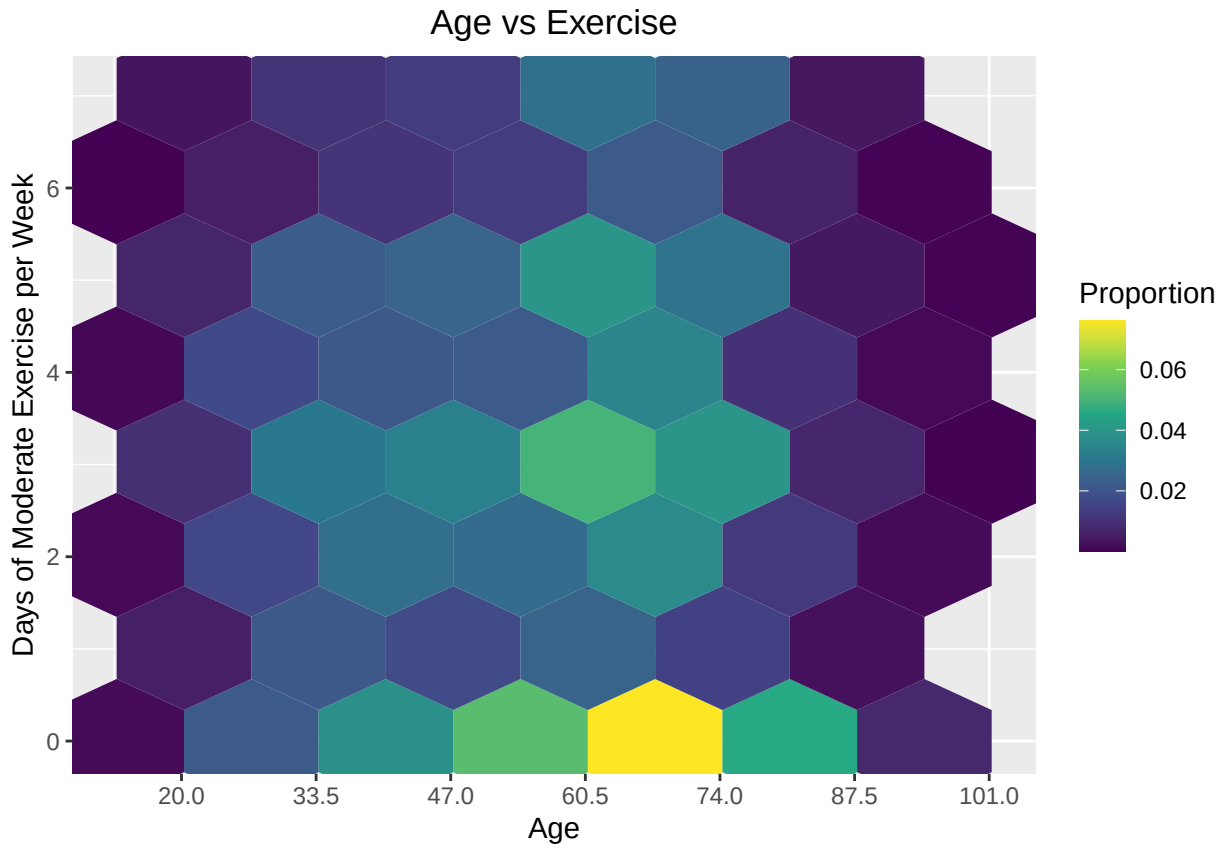
```r
df$household <- as.integer(df$household)
df$exercise <- as.character(df$exercise)
df$exercise <- as.integer(df$exercise)

# reorder columns
df <- df %>%
    select(2, 3, 1, everything())
```

Hexbin Visualization

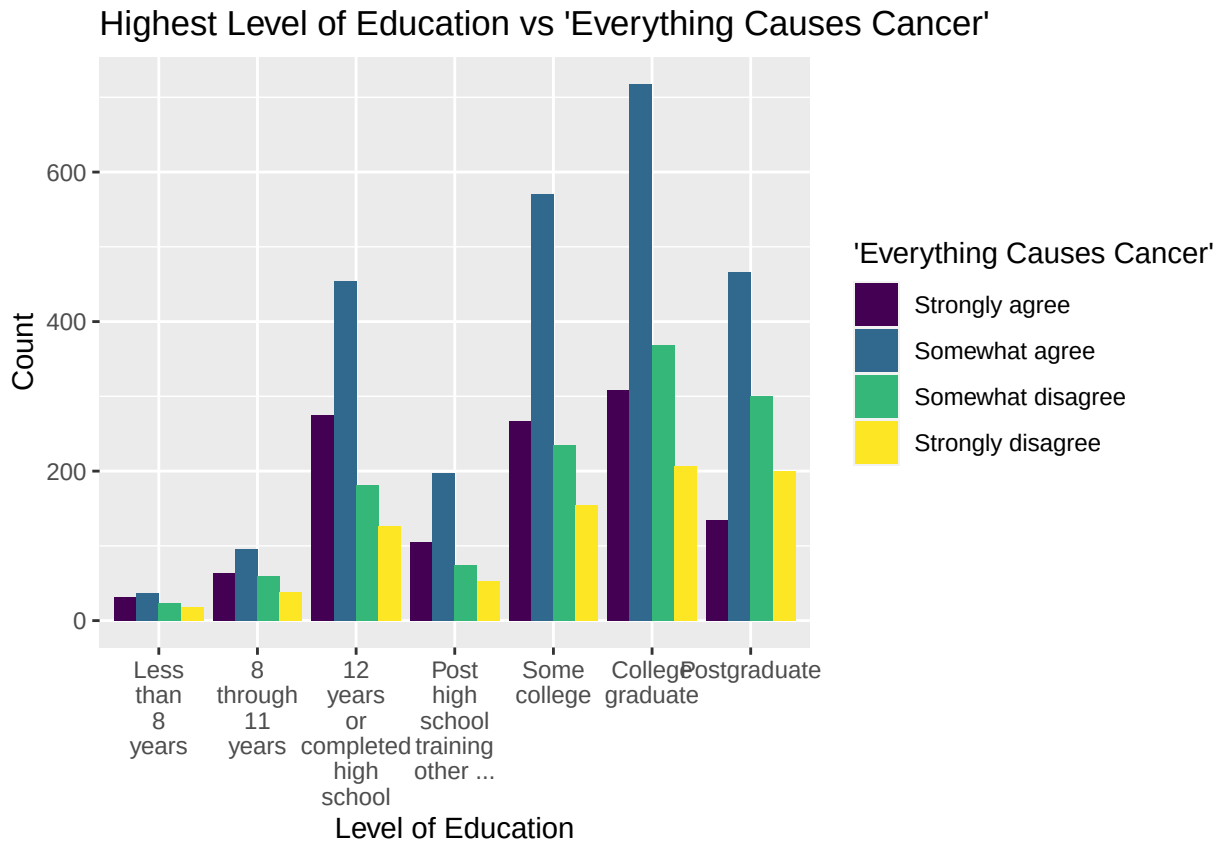
```r
# Hexbin plot for Age vs Exercise
ticks <- seq(min(df$age + 2, na.rm = TRUE), max(df$age + 2, na.rm = TRUE), length.out = 7)
hbData <- df %>%
  filter(!is.na(age), !is.na(exercise))

ggplot(hbData, aes(x = age, y = exercise)) +
  stat_binhex(bins = 6, aes(fill = after_stat(count) / sum(after_stat(count)))) + # bin/total count
  scale_fill_viridis_c() + # color scale
  scale_x_continuous(breaks = ticks) + # custom tick marks
  xlab("Age") +
  ylab("Days of Moderate Exercise per Week") +
  ggtitle("Age vs Exercise") +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(fill = "Proportion") # legend title
```



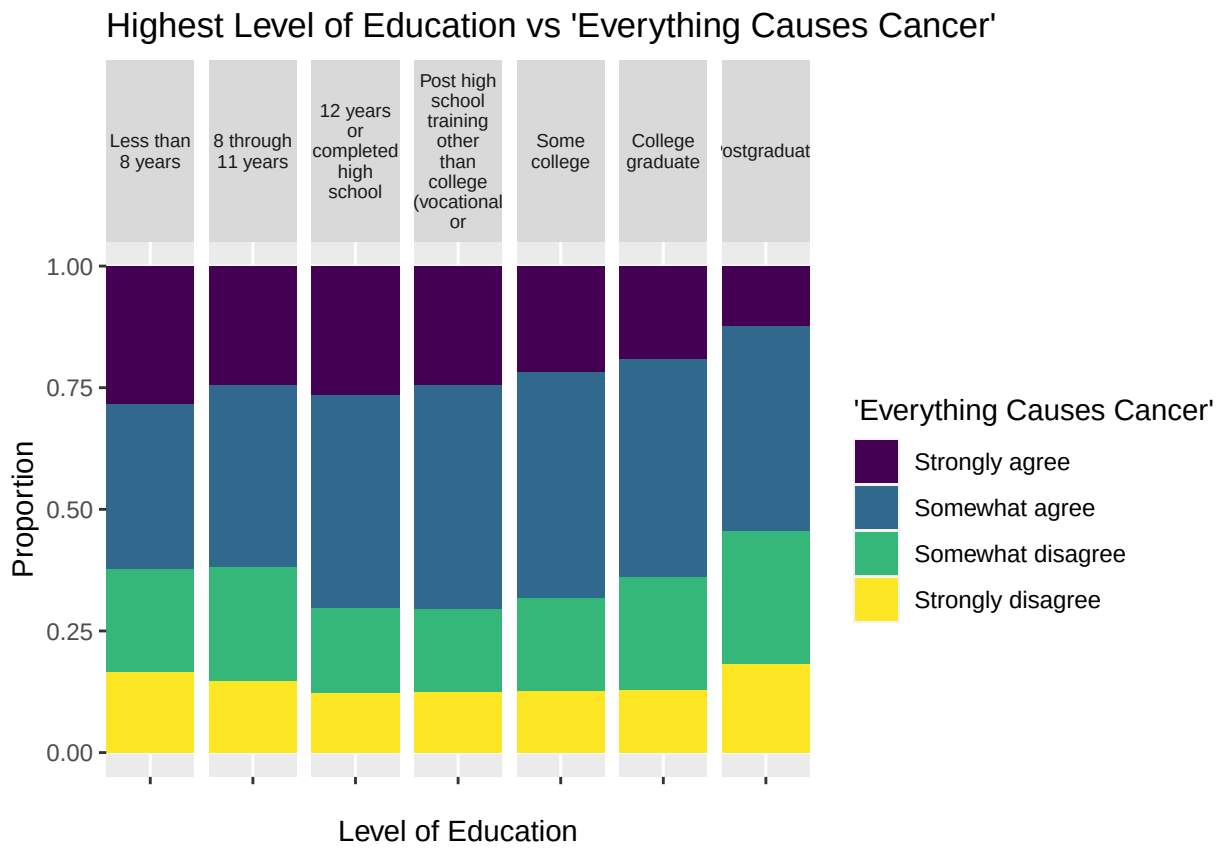
Bar Chart Visualizations

```r
# Bar Chart
ggplot(subset(df, complete.cases(education, cancerEverything)), aes(x = factor(education, exclude = NULL
  geom_bar(position = "dodge") + # each education level has bars for each agreeance level
  scale_fill_viridis_d() + # color scale
  xlab("Level of Education") +
  ylab("Count") +
  ggtitle("Highest Level of Education vs 'Everything Causes Cancer'") +
  scale_x_discrete(labels = function(x) str_wrap(str_trunc(x, width = 35), width = 1)) + #label cut
  labs(fill = "'Everything Causes Cancer'") # legend title
```




```r
# Stacked
sData <- df %>%
  filter(complete.cases(education, cancerEverything)) %>% # filter out NAs
  count(education, cancerEverything, name = "count") # pairwise counts 28x3 tibble

ggplot(sData, aes(x = "", y = count, fill = cancerEverything)) +
  geom_col(position = "fill", width = 1.2) + # y axis is proportions, widen bars a little
  facet_grid(.~education, labeller = label_wrap_gen(width = 10)) + # education levels, label width
  scale_fill_viridis_d() + # color scale
  xlab("Level of Education") +
  ylab("Proportion") +
  ggtitle("Highest Level of Education vs 'Everything Causes Cancer'") +
  theme(strip.text = element_text(size = 7)) + # shrink education labels
  labs(fill = "'Everything Causes Cancer'") # legend title
```

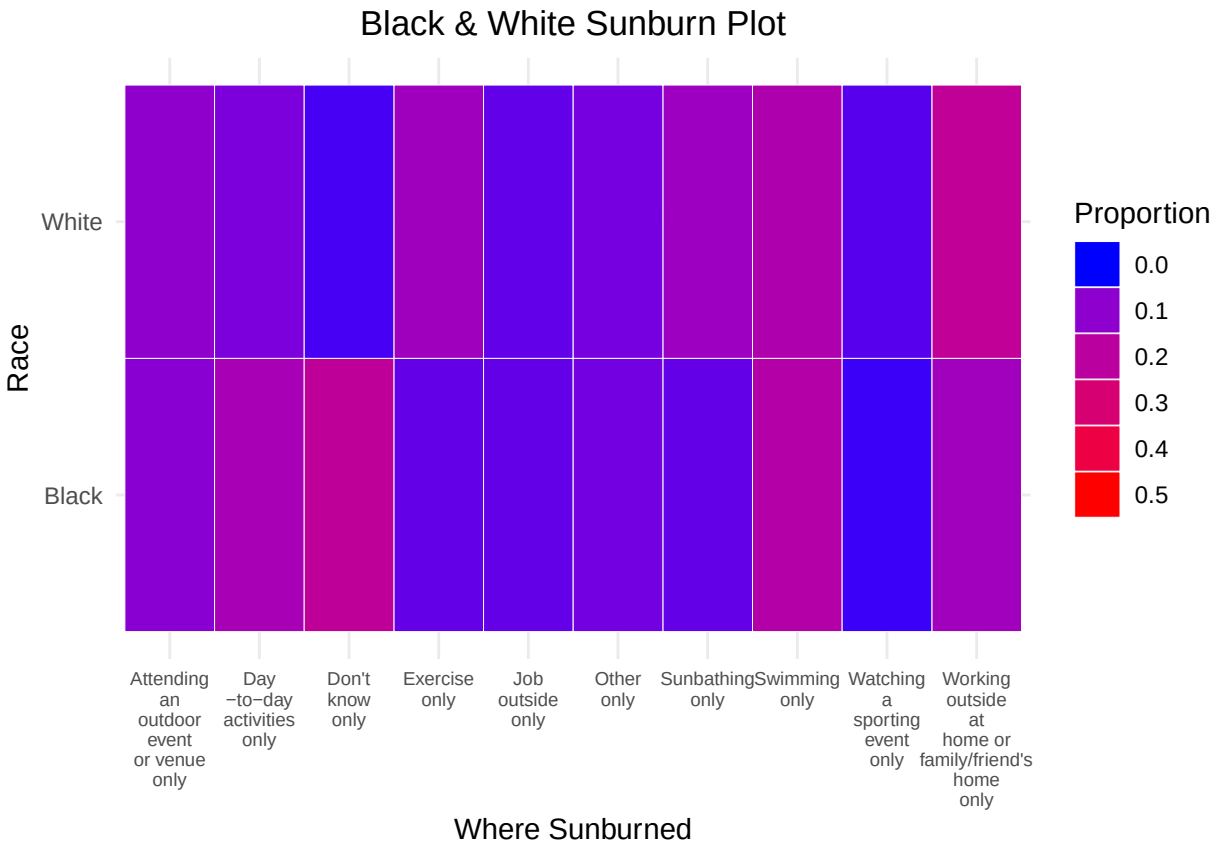


Association Plot

```
# Association Plot for where someone was most recently sunburned (just Black or just White only)
dfRaceSun <- df %>%
  filter(!is.na(race), !is.na(sunburned))
counts <- table(dfRaceSun$race, dfRaceSun$sunburned) # produces contigency table in matrix form
counts <- counts[ ,-c(1:5)] # removes relationships where sunburned data is missing
dfCounts <- as.data.frame.matrix(counts) # turns matrix into data frame

dfBW <- dfCounts[c(5, 17), ] # extracts Black only and White only data to create 2x11 data frame
dfBW <- dfBW %>%
  rownames_to_column(var = "Group") %>% # row names are alternated into a new column named Group
  gather(key = "Variable", value = "Count", -Group) # each column from dfCounts added as a value in "Va
# each value at the intersection is added to a new "Count" column
dfBW <- dfBW[-c(19,20), ] # removes "Multiple Activities Selected" data
dfBW <- dfBW %>%
  group_by(Group) %>% # groups data by black and white to calculate proportions for each group in next
  mutate(Proportion = Count / sum(Count)) # adds a column that calculates the proportion for each sunbu

ggplot(dfBW, aes(x = Variable, y = Group, fill = Proportion)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "blue", high = "red", guide = "legend", limits = c(0, .5)) + # proportion c
  labs(title = "Black & White Sunburn Plot", x = "Where Sunburned", y = "Race") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5), axis.text.x = element_text(size = 7)) + # shrink sunbur
  scale_x_discrete(labels = function(x) str_wrap(x, width = 8)) # wrap sunburn label text
```


Black & White Sunburn Plot
Race
White
Black
Attending an outdoor event or venue only
Day −to−day activities only
Don't know only
Exercise only
Job outside only
Other only
Sunbathing only
Swimming only
Watching a sporting event only
Working outside at home or family/friend's home only
Where Sunburned
Proportion
0.0
0.1
0.2
0.3
0.4
0.5