Statistics 506- Problem Set #3

Garrett Pinkston

Link to GitHub

github: https://github.com/garrettpinkston2015/Computational-Methods

Problem 1 - Vision

a) Download the file VIX_D from this location, and determine how to read it into R. Then download the file DEMO_D from this location. Note that each page contains a link to a documentation file for that data set. Merge the two files to create a single data.frame, using the SEQN variable for merging. Keep only records which matched. Print out your total sample size, showing that it is now 6,980.

```
# used haven library to read xpt files
# https://haven.tidyverse.org/reference/read_xpt.html

library(haven)

#read in vix data
vixData <- read_xpt("/Users/garrettpinkston/Desktop/Michigan/STAT506/Data/VIX_D.XPT")

#read in demo data
demoData <- read_xpt("/Users/garrettpinkston/Desktop/Michigan/STAT506/Data/DEMO_D.XPT")

#merge both dataframes
df <- merge(vixData, demoData, by = "SEQN", all = FALSE)

# remap categorical coding
df$VIQ220[df$VIQ220 == 1] = 0
df$VIQ220[df$VIQ220 == 2] = 1

#if data is 9 or . we can assume it is NaN
```

```r
df$VIQ220[df$VIQ220 == 9] = NA
df$VIQ220[df$VIQ220 == "."] = NA


# Checking the total rows here
nsamples <- nrow(df)
print(nsamples)
```

```
[1] 6980
```

b) Without fitting any models, estimate the proportion of respondents within each 10-year age bracket (e.g. 0-9, 10-19, 20-29, etc) who wear glasses/contact lenses for distance vision. Produce a nice table with the results.

```r
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
# RIDAGEYR is age variable
# VIQ220 is glasses variable

# drop all isnans in data for RIDAGEYR (age) (identified earlier)
df <- df[!is.na(df$RIDAGEMN), ]

# drop all isnans in data for VIQ220 (glasses) (identified earlier)
df <- df[!is.na(df$VIQ220), ]

# create 10 year age groups
df$age_group <- floor(df$RIDAGEYR / 10)
```

```r
# define age group as the range + 9
df$age_group <- paste0(df$age_group * 10, "-", (df$age_group * 10 + 9))




# Calculate proportion of people who wear glasses/contacts by age group
wear_glasses <- df %>%
  # group everyone together by age group. For every age group do the following:
  group_by(age_group) %>%
  # return the following in summary format
  summarize(
    # total is the number of rows
    total = n(),

    # glasses wearers is just summing over all instances where 1's wear glasses
    wear_glasses = sum(VIQ220 == 1, na.rm = TRUE),

    #proportion is the number of glass wearers / total people
    proportion = wear_glasses / total
  )

#using knitr library for output
library(knitr)

#using kable function to output following in a "nice" table
kable(wear_glasses,

      # set caption for graph here
      caption = "Respondents by Age Group",

      #specify names for columns here
      col.names = c("Age Range", "Total Respondents", "Wear Glasses",
                    "Proportion Wearing Glasses"),

      #specify precision here
      digits = 3)
```

Table 1: Respondents by Age Group

Age Range	Total Respondents	Wear Glasses	Proportion Wearing Glasses
10-19	2088	1418	0.679
20-29	937	631	0.673
30-39	750	481	0.641
40-49	773	487	0.630
50-59	609	274	0.450
60-69	630	238	0.378
70-79	447	148	0.331
80-89	188	63	0.335

c) Fit three logistic regression models predicting whether a respondent wears glasses/contact lenses for distance vision. Predictors:

```
# clean up RIAGENDER column and remap 2's to 1's
df$RIAGENDR <- ifelse(df$RIAGENDR == 2, 1, 0)

#drop all NaNs in RIAGENDER column
df$RIAGENDR[is.na(df$RIAGENDR)] <- NA


# make RIDRETH (race) factor into 5 labels (better for description)
df$RIDRETH1 <- factor(df$RIDRETH1,
                      levels = c(1, 2, 3, 4, 5),
                      labels = c("Mexican American", "Other Hispanic",
                                 "Non-Hispanic White", "Non-Hispanic Black",
                                 "Multi-racial"))
```

i) age

```
# double check glasses either is 1 or 0
df$glasses <- ifelse(df$VIQ220 == 1, 1, 0)

# fit model 1
model1 <- glm(glasses ~ RIDAGEYR, data = df, family = binomial)
```

ii) age, race, gender

```r
# add in race and gender into model 2, save into new output
model2 <- glm(glasses ~ RIDAGEYR + RIAGENDR + RIDRETH1, data = df, family = binomial)
```

iii) age, race, gender, Poverty Income ratio

```r
# add in race, gender and PIR into model 3, save into new output

model3 <- glm(glasses ~ RIDAGEYR + RIAGENDR + RIDRETH1 + INDFMPIR, data = df, family = binom
```

Produce a table presenting the estimated odds ratios for the coefficients in each model, along with the sample size for the model, the pseudo- , and AIC values.

```r
#' Get Model Statistics
#'
#' This function extracts key statistics from a fitted logistic regression model.
#' It returns the odds ratios, sample size, pseudo-R^2, and AIC for the model.
#'
#' @param model A fitted logistic regression model object (e.g., from `glm`
#' # with family `binomial`).
#'
#' @return A list containing the following elements:
#' \describe{
#'   \item{or}{Odds ratios (exponentiated coefficients) from the logistic
#'   regression model.}
#'
#'   \item{n}{Sample size (number of observations) used in the model.}
#'
#'   \item{pseudo_r2}{Pseudo-R^2, which is a measure of goodness-of-fit for
#'   logistic regression models.}
#'
#'   \item{aic}{Akaike Information Criterion (AIC), a measure of the relative
#'   quality of the model.}
#' }
#' @export
#'
#' @examples
#' # Example using a logistic regression model
#' model <- glm(outcome ~ predictor1 + predictor2, family = binomial,
#' data = your_data)
#'
#' stats <- get_model_stats(model)
```

```r
#' print(stats)
get_model_stats <- function(model) {
  # take odds ratio first as the exponentiated coefficients from logistic
  # regression model
  or <- exp(coef(model)) # Odds ratios

  # take sample size as the number of observations fed into the model
  n <- nobs(model)       # Sample size

  # Compute  pseudo-R^2 by subtracting the ratio of the model's deviance to the
  # null deviance from 1.

  pseudo_r2 <- 1 - model$deviance / model$null.deviance # Pseudo-R^2

  #calculate AIC from the given model
  aic <- AIC(model)      # AIC

  # return a list of the odds ratio, sample size, pseudor2, aic
  list(or = or, n = n, pseudo_r2 = pseudo_r2, aic = aic)
}
```

```r
# call this function across all models
model1Stats <- get_model_stats(model1)
model2Stats <- get_model_stats(model2)
model3Stats <- get_model_stats(model3)

# Ensure the number of rows in each section is the same by repeating
# model name accordingly
table <- data.frame(
  # Create the 'Model' column by repeating "M1", "M2", and "M3" for the
  # length of each model's odds ratios
  Model = c(rep("M1", length(model1Stats$or)),
            rep("M2", length(model2Stats$or)),
            rep("M3", length(model3Stats$or))),

  # Create the 'Coefficient' column using the names of the odds ratios for each model
  Coefficient = c(names(model1Stats$or), names(model2Stats$or), names(model3Stats$or)),

  # Create the 'Odds_Ratios' column by combining odds ratios from all three models
  Odds_Ratios = c(model1Stats$or, model2Stats$or, model3Stats$or),

  # Create the 'Sample_Size' column by repeating each model's sample size
```

```
  # for the length of its odds ratios
  Sample_Size = c(rep(model1Stats$n, length(model1Stats$or)),
                  rep(model2Stats$n, length(model2Stats$or)),
                  rep(model3Stats$n, length(model3Stats$or))),

  # Create the 'Pseudo_R2' column by repeating each model's pseudo-R^2
  # for the length of its odds ratios
  Pseudo_R2 = c(rep(model1Stats$pseudo_r2, length(model1Stats$or)),
                rep(model2Stats$pseudo_r2, length(model2Stats$or)),
                rep(model3Stats$pseudo_r2, length(model3Stats$or))),

  # Create the 'AIC' column by repeating each model's AIC
  # for the length of its odds ratios
  AIC = c(rep(model1Stats$aic, length(model1Stats$or)),
          rep(model2Stats$aic, length(model2Stats$or)),
          rep(model3Stats$aic, length(model3Stats$or)))
)

# print the ouput
print(table)
```

```
   Model                Coefficient Odds_Ratios Sample_Size  Pseudo_R2      AIC
1     M1                (Intercept)   3.5502108        6422 0.04712484 8320.393
2     M1                   RIDAGEYR   0.9754244        6422 0.04712484 8320.393
3     M2                (Intercept)   6.4353812        6422 0.07001346 8130.628
4     M2                   RIDAGEYR   0.9772122        6422 0.07001346 8130.628
5     M2                   RIAGENDR   0.5990787        6422 0.07001346 8130.628
6     M2     RIDRETH1Other Hispanic   0.8527499        6422 0.07001346 8130.628
7     M2 RIDRETH1Non-Hispanic White   0.5102812        6422 0.07001346 8130.628
8     M2 RIDRETH1Non-Hispanic Black   0.7609945        6422 0.07001346 8130.628
9     M2       RIDRETH1Multi-racial   0.5117459        6422 0.07001346 8130.628
10    M3                (Intercept)   7.6973998        6136 0.07193538 7766.303
11    M3                   RIDAGEYR   0.9777162        6136 0.07193538 7766.303
12    M3                   RIAGENDR   0.5922617        6136 0.07193538 7766.303
13    M3     RIDRETH1Other Hispanic   0.8878480        6136 0.07193538 7766.303
14    M3 RIDRETH1Non-Hispanic White   0.6060632        6136 0.07193538 7766.303
15    M3 RIDRETH1Non-Hispanic Black   0.8030748        6136 0.07193538 7766.303
16    M3       RIDRETH1Multi-racial   0.5762278        6136 0.07193538 7766.303
17    M3                   INDFMPIR   0.8903520        6136 0.07193538 7766.303
```

d) From the third model from the previous part, test whether the odds of men and women being wears of glasess/contact lenses for distance vision differs. Test whether the proportion of

wearers of glasses/contact lenses for distance vision differs between men and women. Include the results of the each test and their interpretation.

```
# Summarize the model to get the coefficient for gender (RIAGENDR)
summary(model3)
```

```
Call:
glm(formula = glasses ~ RIDAGEYR + RIAGENDR + RIDRETH1 + INDFMPIR,
    family = binomial, data = df)

Coefficients:
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)                 2.040883   0.088812  22.980  < 2e-16 ***
RIDAGEYR                   -0.022536   0.001348 -16.715  < 2e-16 ***
RIAGENDR                   -0.523807   0.054853  -9.549  < 2e-16 ***
RIDRETH1Other Hispanic     -0.118955   0.168489  -0.706  0.48018
RIDRETH1Non-Hispanic White -0.500771   0.075692  -6.616 3.69e-11 ***
RIDRETH1Non-Hispanic Black -0.219307   0.079550  -2.757  0.00584 **
RIDRETH1Multi-racial       -0.551252   0.140973  -3.910 9.22e-05 ***
INDFMPIR                   -0.116138   0.017876  -6.497 8.20e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 8351.0  on 6135  degrees of freedom
Residual deviance: 7750.3  on 6128  degrees of freedom
  (286 observations deleted due to missingness)
AIC: 7766.3

Number of Fisher Scoring iterations: 4
```

```
# Extract the odds ratio for gender (exponentiating the coefficient from model)
odds_ratio_gender <- exp(coef(model3)["RIAGENDR"])

# Print the odds ratio
print(paste("The odds ratio for gender (men compared to women) is:", odds_ratio_gender))
```

```
[1] "The odds ratio for gender (men compared to women) is: 0.59226167364357"
```

Based on the summary output from the model, we can tell that the p-value of the is less than 2e^-16, meaning it is a very significant predictor for the model. Therefore, we can reject the null hypothesis that the odds of wearing glasses/contact lenses are the same for men and women. The odds ratio tell us the factor by which the odds change for men compared to women.

```
# Create contingency table for gender and glasses/contact lenses usage
table_gender_glasses <- table(df$RIAGENDR, df$glasses)

# Perform a chi-squared test for men and women
chi_square_test <- chisq.test(table_gender_glasses)

# print the results of the test
print(chi_square_test)
```

```
    Pearson's Chi-squared test with Yates' continuity correction

data:  table_gender_glasses
X-squared = 70.05, df = 1, p-value < 2.2e-16
```

The chi-square test has compared the proportions of glasses/contact lenses wearers between men and women. Because the p-value from the chi-square test is less than 0.05, we can reject the null hypothesis and confidently conclude that the proportion of glasses/contact lenses wearers is different for men and women.

Problem 2 - Sakila

Load the "sakila" database discussed in class into SQLite. It can be downloaded from https://github.com/bradleygrant/sakila-sqlite3.

For these problems, do not use any of the tables whose names end in _list.

```
library(RSQLite)
library(DBI)

#specify path of the database
db_path <- "/Users/garrettpinkston/Desktop/Michigan/STAT506/Data/sakila_master.db"

# Connect to the SQLite database
con <- dbConnect(RSQLite::SQLite(), dbname = db_path)
```

a) What is the oldest movie (earliest release year) in the database? Answer this with a single SQL query.

```
# SQL query to find the oldest movie year and the number of movies released
# in that year
query <- "
  -- Select the release year and count the number of movies for each year
  SELECT release_year, COUNT(*) AS movie_count
  FROM film

  -- Ensure only rows where release_year is not NULL are included
  WHERE release_year IS NOT NULL

  -- Group the data by release year to count movies per year
  GROUP BY release_year

  -- Sort the results by release year in ascending order
  ORDER BY release_year ASC

  -- Limit the output to the first row (i.e., the earliest release year)
  LIMIT 1;
"

# Execute the query and get the result
oldest_movie <- dbGetQuery(con, query)

# Display the result
print(oldest_movie)
```

```
  release_year movie_count
1         2006        1000
```

For each of the following questions, solve them in two ways: First, use SQL query or queries to extract the appropriate table(s), then use regular R operations on those data.frames to answer the question. Second, use a single SQL query to answer the question.

b) What genre of movie is the least common in the data, and how many movies are of this genre?

```
# Define the SQL query to get the movie count for each genre
query_genres <- "
  -- Select the genre name from the category table
```

```
  SELECT c.name AS genre, COUNT(f.film_id) AS movie_count

  -- From the film table (f), join the film_category table (fc)
  -- to connect films to their categories
  FROM film f
  JOIN film_category fc ON f.film_id = fc.film_id

  -- Join the category table (c) to access the genre names
  JOIN category c ON fc.category_id = c.category_id

  -- Group the results by genre name to count movies for each genre
  GROUP BY c.name;
"

# Execute the SQL query and store the result in the genres_data dataframe
genres_data <- dbGetQuery(con, query_genres)

# Find the row with the least common genre by identifying the minimum movie count
least_common_genre <- genres_data[which.min(genres_data$movie_count), ]

# Display the least common genre and its movie count
print(least_common_genre)
```

```
   genre movie_count
12 Music          51
```

```
#single query
query_least_common_genre <- "
  SELECT c.name AS genre, COUNT(f.film_id) AS movie_count
  FROM film f
  JOIN film_category fc ON f.film_id = fc.film_id
  JOIN category c ON fc.category_id = c.category_id
  GROUP BY c.name
  ORDER BY movie_count ASC
  LIMIT 1;
"
# Execute the query
least_common_genre <- dbGetQuery(con, query_least_common_genre)

# Display the result
print(least_common_genre)
```

```
  genre movie_count
1 Music          51
```

c) Identify which country or countries have exactly 13 customers.

```
# Define the SQL query to count the number of customers in each country
query_countries <- "
  -- Select the country name and count the number of customers in each country
  SELECT co.country, COUNT(c.customer_id) AS customer_count

  -- From the customer table (c), join the address table (a) using address_id
  -- to associate customers with their addresses
  FROM customer c
  JOIN address a ON c.address_id = a.address_id

  -- Join the city table (ci) using city_id to connect addresses to cities
  JOIN city ci ON a.city_id = ci.city_id

  -- Join the country table (co) using country_id to connect cities to countries
  JOIN country co ON ci.country_id = co.country_id

  -- Group the results by country to count the number of customers per country
  GROUP BY co.country;
"

# Execute the SQL query and store the result in the countries_data dataframe
countries_data <- dbGetQuery(con, query_countries)

# Filter the dataframe to find the country or countries with exactly 13 customers
countries_with_13_customers <- countries_data[countries_data$customer_count == 13, ]

# Display the country or countries with exactly 13 customers
print(countries_with_13_customers)
```

```
     country customer_count
6  Argentina             13
68   Nigeria             13
```

```
query_countries_13_customers <- "
  -- Select the country name and count the number of customers per country
  SELECT co.country, COUNT(c.customer_id) AS customer_count
```

```
  -- From the customer table (c), join the address table (a) using address_id
  -- to associate customers with their addresses
  FROM customer c
  JOIN address a ON c.address_id = a.address_id

  -- Join the city table (ci) using city_id to connect addresses to cities
  JOIN city ci ON a.city_id = ci.city_id

  -- Join the country table (co) using country_id to connect cities to countries
  JOIN country co ON ci.country_id = co.country_id

  -- Group the results by country to count the number of customers per country
  GROUP BY co.country

  -- Use HAVING to filter the grouped results to only include countries
  -- with exactly 13 customers
  HAVING customer_count = 13;
"

# Execute the SQL query and store the result into dataframe
countries_with_13_customers <- dbGetQuery(con, query_countries_13_customers)

# Display the country or countries that have exactly 13 customers
print(countries_with_13_customers)
```

```
    country customer_count
1 Argentina             13
2   Nigeria             13
```

Problem 3 - US Records

Download the "US - 500 Records" data from https://www.briandunning.com/sample-data/ and import it into R. This is entirely fake data - use it to answer the following questions.

```
# read in data
df <- read.csv("/Users/garrettpinkston/Desktop/Michigan/STAT506/Data/us-500.csv")
```

a) What proportion of email addresses are hosted at a domain with TLD ".com"? (in the email, "angrycat@freemail.org", "freemail.org" is the domain, and ".org" is the TLD (top-level domain).)

```r
# Calculate the proportion of emails ending with "com" in the 'email' column of 'df'
length(df$email[grepl("com$", df$email)])/nrow(df)
```

```
[1] 0.732
```

b) What proportion of email addresses have at least one non alphanumeric character in them? (Excluding the required "@" and "." found in every email address.)

```r
# split email addresses into two parts (splitting at "@")
emails <- strsplit(df$email, "@")

# Extract the username part of each email (part before "@")
usernames <- sapply(emails, "[[", 1)

# Check if each username contains any non-alphanumeric character
# (excluding letters and digits)
username_non_alphanumeric <- grepl("[^a-zA-Z0-9]", usernames)


# Extract the domain part of each email (the part after "@")
domains <- sapply(emails, "[[", 2)

# remove the organization (e.g., ".com", ".net") from the domain part
domains <- gsub("\\.[a-z]{2,}$", "", domains)

# check whether each domain (excluding the top-level domain) contains any
# non-alphanumeric character
domain_non_alphanumeric <- grepl("[^a-zA-Z0-9]", domains)

# Calculate the proportion of emails where either the username or domain
# has a non-alphanumeric character

proportion_non_alphanumeric <- mean(username_non_alphanumeric | domain_non_alphanumeric)

# Output result
proportion_non_alphanumeric
```

```
[1] 0.506
```

c) What are the top 5 most common area codes amongst all phone numbers? (The area code is the first three digits of a standard 10-digit telephone number.)

```r
# Extract the 'phone1' column from the dataframe (phone numbers)
phone_numbers <- df$phone1

# Extract the first 3 digits of each phone number (area codes)
area_codes <- substr(phone_numbers, 1, 3)

# Count the frequency of each area code
area_code_counts <- table(area_codes)

# Sort the area code counts in decreasing order and select the top 5 most common area codes
top_5_area_codes <- sort(area_code_counts, decreasing = TRUE)[1:5]

# Print the top 5 area codes and their counts
print(top_5_area_codes)
```

```
area_codes
973 212 215 410 201
 18  14  14  14  12
```

d) Produce a histogram of the log of the apartment numbers for all addresses. (You may assume any number at the end of the an address is an apartment number.)

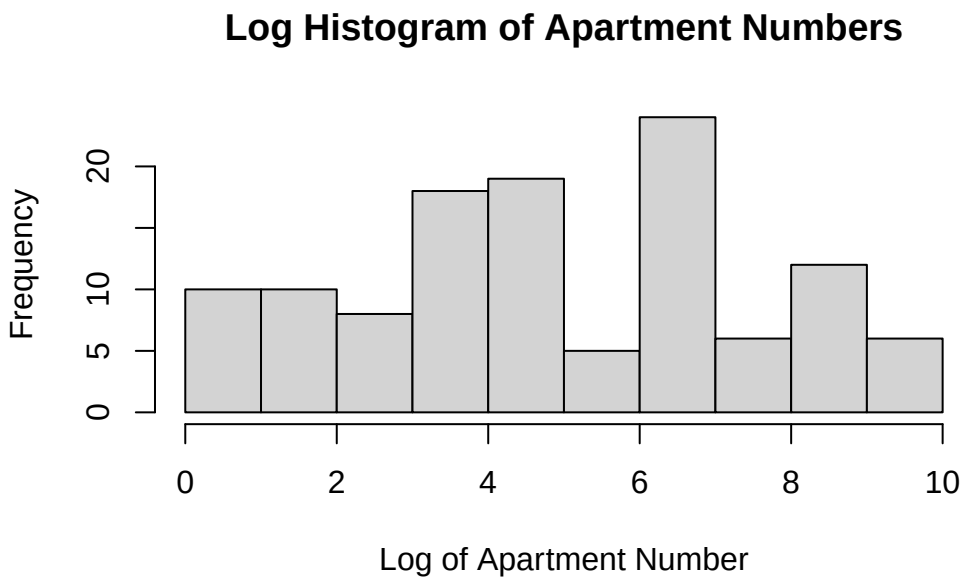
```r
# load ggplot2 for visualization
library(ggplot2)

# Extract addresses that end with a number (typically apartment numbers)
apartments <- df$address[grepl("[0-9]+$", df$address)]

# split addresses into individual words and extract the apartment number
numbers <- sapply(strsplit(apartments, " "), function(x) x[length(x)])

# Remove any '#' characters (if present) from the apartment numbers
# and convert them to numeric values
numbers <- as.numeric(gsub("#", "", numbers))

# Make a histogram of the log-transformed apartment numbers
hist(log(numbers),
     main = "Log Histogram of Apartment Numbers",
     xlab = "Log of Apartment Number")
```



e) Benford's law is an observation about the distribution of the leading digit of real numerical data. Examine whether the apartment numbers appear to follow Benford's law. Do you think the apartment numbers would pass as real data?

```
#print table output of the numbers
table(substr(numbers, 1, 1))
```

```
 1  2  3  4  5  6  7  8  9
15 13 12 12 15 11 12 11 17
```

Based on the output of table(substr(numbers, 1, 1)), the frequencies of the leading digits appear uniform rather than following the expected decreasing distribution from Benford's Law. Benford's Law states that lower digits should appear more frequently than higher digits. The fact that this data is roughly uniform indicates that this data does not represent naturally occurring patterns and suggests that it is synthetic.