ProblemSet1

Yiming Chen

Problem 1 - Abalone Data

a

```
col_names <- c("Sex", "Length", "Diameter", "Height",
               "Whole_weight", "Shucked_weight",
               "Viscera_weight", "Shell_weight", "Rings")

abalone <- read.csv("C:/Users/cyming/Desktop/STATS 506/Assignment/ProblemSet1/abalone/abalone
head(abalone)
```

```
  Sex Length Diameter Height Whole_weight Shucked_weight Viscera_weight
1   M  0.455    0.365  0.095       0.5140         0.2245         0.1010
2   M  0.350    0.265  0.090       0.2255         0.0995         0.0485
3   F  0.530    0.420  0.135       0.6770         0.2565         0.1415
4   M  0.440    0.365  0.125       0.5160         0.2155         0.1140
5   I  0.330    0.255  0.080       0.2050         0.0895         0.0395
6   I  0.425    0.300  0.095       0.3515         0.1410         0.0775
  Shell_weight Rings
1        0.150    15
2        0.070     7
3        0.210     9
4        0.155    10
5        0.055     7
6        0.120     8
```

b

```
table(abalone$Sex)
```

```

   F    I    M 
1307 1342 1528 
```

There area 1307 female abalones, 1528 male abalones and 1342 infant abalones.

c

```
weights <- abalone[, c("Whole_weight", "Shucked_weight",
                       "Viscera_weight", "Shell_weight")]

cor_with_rings <- cor(weights, abalone$Rings)
cor_with_rings
```

```
                    [,1]
Whole_weight   0.5403897
Shucked_weight 0.4208837
Viscera_weight 0.5038192
Shell_weight   0.6275740
```

1. Shell weight has the highest correlation with rings.

```
by(abalone, abalone$Sex, function(df) cor(df$Shell_weight, df$Rings))
```

```
abalone$Sex: F
[1] 0.405907
------------------------------------------------------------ 
abalone$Sex: I
[1] 0.7254357
------------------------------------------------------------ 
abalone$Sex: M
[1] 0.5109967
```

2. For this weight, infant has the highest correlation with rings.

```
max_rings <- max(abalone$Rings)
subset(abalone, Rings == max_rings)[, c("Whole_weight", "Shucked_weight", "Viscera_weight", "
```

```
    Whole_weight Shucked_weight Viscera_weight Shell_weight
481       1.8075         0.7055         0.3215        0.475
```

3. The most rings abalone’s whole weight is 1.8075 grams, its shucked weight is 0.7055 grams, viscera weight is 0.3215 grams and its shell weight is 0.475 grams.

```
mean(abalone$Viscera_weight > abalone$Shell_weight) * 100
```

```
[1] 6.511851
```

4. 6.511851% of abalones have a viscera weight larger than their shell weight.

d

```
sexes <- unique(abalone$Sex)

cor_table <- sapply(sexes, function(s) {
  df <- subset(abalone, Sex == s)
  cor(df[, c("Whole_weight","Shucked_weight","Viscera_weight","Shell_weight")], df$Rings)
})

cor_table <- t(cor_table)
rownames(cor_table) <- sexes
colnames(cor_table) <- c("Whole_weight","Shucked_weight","Viscera_weight","Shell_weight")
cor_table
```

```
  Whole_weight Shucked_weight Viscera_weight Shell_weight
M    0.3721966     0.22239382      0.3209535    0.5109967
F    0.2667585     0.09484802      0.2116154    0.4059070
I    0.6963268     0.62024577      0.6732727    0.7254357
```

The table of correlation is shown above.

e

```
t.test(Rings ~ Sex, data = subset(abalone, Sex %in% c("M","F")))
```

```

	Welch Two Sample t-test

data:  Rings by Sex
t = 3.6657, df = 2742.4, p-value = 0.0002514
alternative hypothesis: true difference in means between group F and group M is not equal to
95 percent confidence interval:
 0.1971045 0.6505082
sample estimates:
mean in group F mean in group M 
        11.1293         10.7055 
```

```
t.test(Rings ~ Sex, data = subset(abalone, Sex %in% c("M","I")))
```

```

	Welch Two Sample t-test

data:  Rings by Sex
t = -27.221, df = 2859, p-value < 2.2e-16
alternative hypothesis: true difference in means between group I and group M is not equal to
95 percent confidence interval:
 -3.017808 -2.612263
sample estimates:
mean in group I mean in group M 
       7.890462       10.705497 
```

```
t.test(Rings ~ Sex, data = subset(abalone, Sex %in% c("F","I")))
```

```

	Welch Two Sample t-test

data:  Rings by Sex
t = 29.477, df = 2508.9, p-value < 2.2e-16
alternative hypothesis: true difference in means between group F and group I is not equal to
95 percent confidence interval:
 3.023380 3.454304
sample estimates:
mean in group F mean in group I 
      11.129304        7.890462 
```

The number of rings differs across the three sexes. I did 3 Welch two sample t-test to examine if the mean number of rings differs across the 3 sexes. For the first t-test between F and M, we can see Females have a slightly higher mean number of rings than males, and this difference is statistically significant due to the very small p-value, equals to 0.0002514. Moreover, for the rest tests, we compare F and M with I, and obviously the p-value is close to 0, showing highly significant difference between mean number of rings. That's very reasonable since abalones gain rings gradually when they grow. And the general ordering is like F > M > I.

Problem 2 - Food Expenditure Data

a

```
food <- read.csv("C:/Users/cyming/Desktop/STATS 506/Assignment/ProblemSet1/food_expenditure.
```

b

```
# install.packages("janitor")
#ibrary(janitor)
#food <- clean_names(food)
colnames(food) <- c(
  "id",
  "age",
  "household_size",
  "state",
  "currency",
  "total_food_exp",
  "grocery_exp",
  "dining_out_exp",
  "misc_food_exp",
  "dine_out_times",
  "alcohol",
  "food_assistance"
)
names(food)
```

```
 [1] "id"              "age"             "household_size"  "state"
 [5] "currency"        "total_food_exp"  "grocery_exp"     "dining_out_exp"
 [9] "misc_food_exp"   "dine_out_times"  "alcohol"         "food_assistance"
```

The simplified variable names are shown above.

c

```r
n_before <- nrow(food)
food_usd <- subset(food, currency == "USD")
n_after <- nrow(food_usd)

cat("Before filtering:", n_before, "After filtering:", n_after, "\n")
```

```
Before filtering: 262 After filtering: 230
```

d

```r
food_usd <- subset(food_usd, age >= 18 & age <= 90)
```

For the age variable, I only kept respondents between 18 and 90 years old.

e

```r
valid_states <- state.abb
food_usd <- subset(food_usd, state %in% valid_states)
```

For the state variable, I used the built-in state abbreviations in R to compare with these entries, and dropped unmatched rows.

f

```r
food_usd$total_food_exp <- suppressWarnings(as.numeric(food_usd$total_food_exp))
food_usd <- subset(food_usd,
                   !is.na(total_food_exp) & total_food_exp >= 0 &
                   !is.na(grocery_exp) & grocery_exp >= 0 &
                   !is.na(dining_out_exp) & dining_out_exp >= 0 &
                   !is.na(misc_food_exp) & misc_food_exp >= 0)
```

For the 4 variables related to food expenditures, I found that the total food expenditure was character type, so I converted it into numeric type. Then I removed NA values and only kept non-negative rows.

g

```r
food_usd <- subset(food_usd,
                   !is.na(dine_out_times) &
                   dine_out_times >= 0)
```

For the variable related to the number of times dining out, I just removed rows with NA values and negative values.

h

```r
cat("Final number of observations after this cleaning:", nrow(food_usd), "\n")
```

```
Final number of observations after this cleaning: 122
```

Problem 3 - Collatz conjecture

a

```r
nextCollatz <- function(n) {
  if (!is.numeric(n) || length(n) != 1 || n <= 0 || n != as.integer(n)) {
    stop("Input must be a positive integer.")
  }
  if (n %% 2 == 0) {
    return(n / 2)
  } else {
    return(3 * n + 1)
  }
}
```

```r
nextCollatz(5)
```

```
[1] 16
```

```r
nextCollatz(16)
```

```
[1] 8
```

b

```r
collatzSequence <- function(n) {
  if (!is.numeric(n) || length(n) != 1 || n <= 0 || n != as.integer(n)) {
    stop("Input must be a positive integer.")
  }

  seq <- n
  while (tail(seq, 1) != 1) {
    seq <- c(seq, nextCollatz(tail(seq, 1)))
  }

  list(sequence = seq, length = length(seq))
}
```

```r
collatzSequence(5)
```

```
$sequence
[1]  5 16  8  4  2  1

$length
[1] 6
```

```r
collatzSequence(19)
```

```
$sequence
 [1] 19 58 29 88 44 22 11 34 17 52 26 13 40 20 10  5 16  8  4  2  1

$length
[1] 21
```

c

```
start_vals <- 100:500
seq_lengths <- sapply(start_vals, function(x) collatzSequence(x)$length)

shortest_start <- start_vals[which.min(seq_lengths)]
longest_start  <- start_vals[which.max(seq_lengths)]

shortest_seq <- collatzSequence(shortest_start)
longest_seq  <- collatzSequence(longest_start)
```

```
cat("Shortest sequence starts at", shortest_start, "with length", shortest_seq$length, "\n")
```

```
Shortest sequence starts at 128 with length 8
```

```
cat("Longest sequence starts at", longest_start, "with length", longest_seq$length, "\n")
```

```
Longest sequence starts at 327 with length 144
```