

STATS506-Problem Set 1

Shenyi Tang

2024-09-05

Shenyi Tang's GitHub Repo For STATS 506 FA 2024

<https://github.com/shenyi-tang/stats506-computing-methods-and-tools.git>

R Packages

```
library("dplyr")
library("scales")
```

Problem 1 - Wine Data

- a. Import the data into a data.frame in R. Use the information in the `wine.names` file to give appropriate column names.

```
wd <- read.table('wine/wine.data',
                 header = FALSE,
                 sep = ',')
names(wd) <- c("class", "alcohol", "m_acid", "ash",
              "alcalinity", "mg", "t_phenols",
              "flv", "nonflv_phenols", "pac",
              "color", "hue", "od", "proline")
```

- b. The data contains information on three different classes of wine. Check and report that the number of wines with in each class is correct as reported in `wine.names`

```
aggregate(wd$class, by = list(wd$class), length)
```

	Group.1	x
1	1	59
2	2	71
3	3	48

- The number of wines in each class is correct as reported in `wine.names`

c. Use the data to answer the following questions:

1. What is the correlation between alcohol content and color intensity?
2. Which class has the highest correlation? Which has the lowest?
3. What is the alcohol content of the wine with the highest color intensity?
4. What percentage of wines had a higher content of proanthocyanins compare to ash?

```
## Question c.1
cor(wd$alcohol, wd$color)
```

```
[1] 0.5463642
```

```
## Question c.2
# correlation group by class
cor_result <- wd %>% group_by(class) %>% summarise(correlation = cor(alcohol, color))
cor_result
```

```
# A tibble: 3 x 2
  class correlation
  <int>      <dbl>
1     1         0.408
2     2         0.270
3     3         0.350
```

```
# find responding class
highest_class <-
  cor_result$class[which(cor_result$correlation == max(cor_result$correlation))]
lowest_class <-
  cor_result$class[which(cor_result$correlation == min(cor_result$correlation))]

#output
cat('class',
    highest_class,
    'has the highest correlation between alcohol and color intensity', "\n")
```

```
class 1 has the highest correlation between alcohol and color intensity
```

```
cat('class',
    lowest_class,
    'has the lowest correlation between alcohol and color intensity', "\n")
```

```
class 2 has the lowest correlation between alcohol and color intensity
```

```
## Question c.3
alcohol_content <- wd$alcohol[which(wd$color == max(wd$color))]
cat("The alcohol content is", alcohol_content,
    "with the highest color intensity", "\n")
```

The alcohol content is 14.34 with the highest color intensity

```
## Question c.4
percentage <- (wd %>% filter(pac > ash) %>% nrow()) / nrow(wd)
cat(percent(percentage, accuracy = 0.001),
    "of wines had a higher content of proanthocyanins compare to ash.", "\n")
```

8.427% of wines had a higher content of proanthocyanins compare to ash.

- d. Create a table identifying the average value of each variable, providing one row for the overall average, and one row per class with class averages

```
# mean by group
mean1 <- aggregate(wd[,2:ncol(wd)], by = list(wd$class), mean)
names(mean1)[1] <- 'Class'
mean1$class <- as.character(mean1$class)

# overall mean
mean2 <- data.frame(t(apply(wd[,2:ncol(wd)], 2, mean)))
# insert a new col as the first col
mean2 <- mean2 %>% mutate(Class = 'overall') %>% select(Class, everything())

bind_rows(mean1, mean2)
```

	Class	alcohol	m_acid	ash	alcalinity	mg t_phenols	flv
1	1	13.74475	2.010678	2.455593	17.03729	106.33898	2.840169
2	2	12.27873	1.932676	2.244789	20.23803	94.54930	2.258873
3	3	13.15375	3.333750	2.437083	21.41667	99.31250	1.678750
4	overall	13.00062	2.336348	2.366517	19.49494	99.74157	2.295112
		nonflv_phenols	pac	color	hue	od	proline
1		0.2900000	1.899322	5.528305	1.0620339	3.157797	1115.7119
2		0.3636620	1.630282	3.086620	1.0562817	2.785352	519.5070
3		0.4475000	1.153542	7.396250	0.6827083	1.683542	629.8958
4		0.3618539	1.590899	5.058090	0.9574494	2.611685	746.8933

- e. Carry out a series of t-tests to examine whether the level of phenols differs across the three classes. Present the R output and interpret the results.

```
phenols1 <- wd[wd$class == 1, "t_phenols"]
phenols2 <- wd[wd$class == 2, "t_phenols"]
phenols3 <- wd[wd$class == 3, "t_phenols"]

# for class 1 and class 2
t.test(phenols1, phenols2)
```

Welch Two Sample t-test

```
data: phenols1 and phenols2
t = 7.4206, df = 119.14, p-value = 1.889e-11
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.4261870 0.7364055
sample estimates:
mean of x mean of y
 2.840169  2.258873
```

```
# for class 2 and class 3
t.test(phenols2, phenols3)
```

Welch Two Sample t-test

```
data: phenols2 and phenols3
t = 7.0125, df = 116.91, p-value = 1.622e-10
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.4162855 0.7439610
sample estimates:
mean of x mean of y
 2.258873  1.678750
```

```
# for class 1 and class 3
t.test(phenols1, phenols3)
```

Welch Two Sample t-test

```
data: phenols1 and phenols3
t = 17.12, df = 98.356, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 1.026801 1.296038
sample estimates:
mean of x mean of y
 2.840169  1.678750
```

- **Class 1 and Class 2:** The t-value is out of the 95% confidence interval and $p - value < 0.05$. Therefore, the null hypothesis can be rejected, and it can be concluded that the levels of total phenols differ between class 1 wines and class 2 wines.
- **Class 2 and Class 3:** The t-value is out of the 95% confidence interval and $p - value < 0.05$. Therefore, the null hypothesis can be rejected, and it can be concluded that the levels of total phenols differ between class 2 wines and class 3 wines.

- **Class 1 and Class 3:** The t-value is out of the 95% confidence interval and $p - value < 0.05$. Therefore, the null hypothesis can be rejected, and it can be concluded that the levels of total phenols differ between class 1 wines and class 3 wines.

Problem 2 - AskManager.org Data

- Import the data into a data.frame in R. As with the wine data, you may download the data outside of your submission, but importation should take place inside the problem set submission.

```
aam <- read.csv2('AskAManager.csv',
                 sep = ',', header = TRUE, encoding = 'utf-8')
```

- Clean up the variable names. Simplify them.

```
names(aam) <- c('no.', 'timestamp', 'age', 'industry', 'job_title',
               'add_job_title', 'salary', 'compensation', 'currency', 'add_currency',
               'add_income', 'country', 'us_state', 'city', 'overall_years',
               'field_years', 'degree', 'gender', 'race')
```

- Restrict the data to those being paid in US dollars (USD). Show that it worked by confirming the number of observations before and after restricting the data.

```
# before restricting data
nrow(aam)
```

```
[1] 28062
```

```
# after restricting data
aam_usd <- aam[aam$currency == 'USD',]
nrow(aam_usd)
```

```
[1] 23374
```

- Assume no one starts working before age 18. Eliminate any rows for which their age, years of experience in their field, and years of experience total are impossible. Again, confirm the number of observations.

```
#' extract the first value of a string
#
#' @param x, a character
#
#' @return a numeric
#
#' @examples input "18-22" return 18

extract_first_value <- function(x) {
  return(as.numeric(gsub("[^0-9].*", "", x)))
}
```

```

aam2 <- aam %>%
  mutate(
    start_age = extract_first_value(age),
    start_overall_years = extract_first_value(overall_years),
    start_field_years = extract_first_value(field_years)
  )

aam3 <- aam2 %>%
  filter(
    start_age >= 18,
    start_overall_years >= start_field_years,
    start_age >= start_overall_years
  )

nrow(aam3)

```

[1] 27782

- e. A lot of incomes are likely false. Eliminate any rows with extremely low or extremely high salaries. I'll leave the decision of what thresholds to use up to you; you could choose to eliminate only impossible values, or you could restrict the sample to eliminate the extreme values even if they are realistic. You must justify your choice, along with either a cited source or an exploration the data, or some combination.

```

# Thresholds for extreme low and extremely high salaries
# Start with dataset aam3
aam4 <- aam3 %>%
  filter(
    salary >= 29250,
    salary <= 121470
  )

nrow(aam4)

```

[1] 21504

- According to [Occupational Employment and Wage Statistics from U.S. Bureau of Labor Statistics in May 2023](#), among all occupations, annual 10th percentile wage is \$29,050, annual 90th percentile wage is \$121,470.

Problem 3 - Palindromic Numbers

- a. Write function `isPalindromic` that checks if a given positive integer is a palindrome. Be sure to provide a reasonable error on an invalid input. Be sure to document your function
- Input: A positive integer
 - Output: A list with two elements:

- * **isPalindromic**: A logical value indicating if the input is palindromic
- * **reversed**: The input with its digits reversed

```
# @title to check if a positive integer is a palindrome
#'
#' @param x, a positive integer
#'
#' @return1 isPalindromic, a logic to indicate if the input is palindromic
#' @return2 reversed, reverse the input
#'

isPalindromic <- function(x) {
  # verify valid input
  if (is.numeric((x)) == FALSE || x <= 0){
    stop("The input should be a positive integer!")
  }

  if (x %% 10 == 0 && x != 0) {
    return(list(isPalindromic = FALSE, reversed = NA))
  }

  char <- as.character(x)
  # reverse and concat the char
  rev_char <- paste(rev(strsplit(char, NULL)[[1]]), collapse = "")

  result <- list(
    isPalindromic = (char == rev_char),
    reversed = as.numeric(rev_char)
  )

  return(result)
}

# test
isPalindromic(39951)
```

```
$isPalindromic
[1] FALSE
```

```
$reversed
[1] 15993
```

```
# test
isPalindromic(734437)
```

```
$isPalindromic
[1] TRUE
```

```
$reversed  
[1] 734437
```

- b. Create a function `nextPalindrome` that finds the next palindromic number strictly greater than the input. Be sure to provide a reasonable error on an invalid input.
- Input: A positive integer
 - Output: A vector of length 1 with the next palindromic number greater than the input

```
#' @title to find the next palindrome strictly greater than the input  
#'  
#' @param x, a positive integer  
#' @return a vector of length 1  
#'  
nextPalindrome <- function(x) {  
  # verify valid input  
  if (is.numeric((x)) == FALSE || x <= 0){  
    stop("The input should be a positive integer!")  
  }  
  
  num <- x + 1  
  while (isPalindromic(num)$isPalindromic == FALSE) {  
    num = num + 1  
  }  
  
  return(num)  
}  
  
# test  
nextPalindrome(7152)
```

```
[1] 7227
```

```
# test  
nextPalindrome(765431537)
```

```
[1] 765434567
```

- c. Use these functions to find the next palindrome for each of the following:

1. 391
2. 9928
3. 19272719
4. 109
5. 2


```
nextPalindrome(391)
```

```
[1] 393
```

```
nextPalindrome(9928)
```

```
[1] 9999
```

```
nextPalindrome(19272719)
```

```
[1] 19277291
```

```
nextPalindrome(109)
```

```
[1] 111
```

```
nextPalindrome(2)
```

```
[1] 3
```