hw1-stats506-lijiabao

JiabaoLi

link of github: https://github.com/lijiabao203/stats506_rwork

doc: hw1-stats506-lijiabao.qmd and hw1-stats506-lijiabao.pdf

problem 1 - wine data

a: Import the data and label the data

Use function "read.table" to read "wine.data" and use "," as attribute "sep" to correctly read.

Save it as data_wine in format "data.frame".

Finally use function "names" to label the data.

```
data_wine = data.frame(read.table("./wine/wine.data", sep = ','))
names(data_wine) = c("Class", "Alcohol", "Malic acid", "Ash", "Alcalinity of ash", "Magnesiu
```

b: Check the number of wines within each class

Firstly use "data_wine["Class"]" to get all classes of wine.

And use function "table" to check the number of wines within each class.

Compared with the number from wine.names, the number we got from R is correct.

```
table(data_wine["Class"])
```

```
Class
 1  2  3
59 71 48
```

c: problem sets

1 What is the correlation between alcohol content and color intensity?

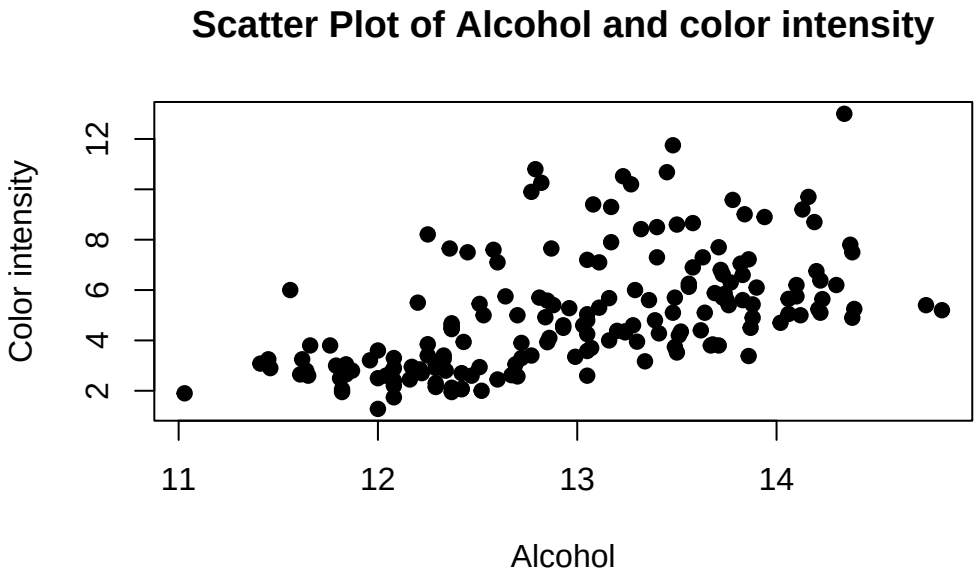
From the table below, alcohol content is always bigger than color intensity.

Besides, we can use function "cor" to compute the correlation coefficient, which is 0.5463642. And we can use a scatter plot to verify the correlation. In fact, there is no direct relationship between these two attribute.

```
c("mean_of_Alcohol" = mean((data_wine["Alcohol"])[[1]]), "mean_of_color_intensity" = mean((da
```

```
        mean_of_Alcohol mean_of_color_intensity correlation_coefficient
             13.0006180               5.0580899               0.5463642
```

```
plot((data_wine["Alcohol"])[[1]], (data_wine["Color intensity"])[[1]], main="Scatter Plot of
abline((data_wine["Alcohol"])[[1]], (data_wine["Color intensity"])[[1]])
```




```
cor_test = cor.test((data_wine["Alcohol"])[[1]], (data_wine["Color intensity"])[[1]])
print(cor_test)
```

```
    Pearson's product-moment correlation

data:  (data_wine["Alcohol"])[[1]] and (data_wine["Color intensity"])[[1]]
```

```
t = 8.6542, df = 176, p-value = 3.056e-15
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.4341702 0.6418673
sample estimates:
      cor
0.5463642
```

2 Which class has the highest correlation? Which has the lowest?

Use function cor to get the classes with highest and lowest classes:

```
cor(data_wine)
```

```
                                      Class     Alcohol  Malic acid          Ash
Class                          1.00000000 -0.32822194  0.43777620 -0.049643221
Alcohol                       -0.32822194  1.00000000  0.09439694  0.211544596
Malic acid                     0.43777620  0.09439694  1.00000000  0.164045470
Ash                           -0.04964322  0.21154460  0.16404547  1.000000000
Alcalinity of ash              0.51785911 -0.31023514  0.28850040  0.443367187
Magnesium                     -0.20917939  0.27079823 -0.05457510  0.286586691
Total phenols                 -0.71916334  0.28910112 -0.33516700  0.128979538
Flavanoids                    -0.84749754  0.23681493 -0.41100659  0.115077279
Nonflavanoid phenols           0.48910916 -0.15592947  0.29297713  0.186230446
Proanthocyanins               -0.49912982  0.13669791 -0.22074619  0.009651935
Color intensity                0.26566757  0.54636420  0.24898534  0.258887259
Hue                           -0.61736921 -0.07174720 -0.56129569 -0.074666889
OD280/OD315 of diluted wines  -0.78822959  0.07234319 -0.36871043  0.003911231
Proline                       -0.63371678  0.64372004 -0.19201056  0.223626264
                              Alcalinity of ash   Magnesium Total phenols
Class                              0.51785911 -0.20917939   -0.71916334
Alcohol                           -0.31023514  0.27079823    0.28910112
Malic acid                         0.28850040 -0.05457510   -0.33516700
Ash                                0.44336719  0.28658669    0.12897954
Alcalinity of ash                  1.00000000 -0.08333309   -0.32111332
Magnesium                         -0.08333309  1.00000000    0.21440123
Total phenols                     -0.32111332  0.21440123    1.00000000
Flavanoids                        -0.35136986  0.19578377    0.86456350
Nonflavanoid phenols               0.36192172 -0.25629405   -0.44993530
Proanthocyanins                   -0.19732684  0.23644061    0.61241308
Color intensity                    0.01873198  0.19995001   -0.05513642
Hue                               -0.27395522  0.05539820    0.43368134
```

```
OD280/OD315 of diluted wines       -0.27676855  0.06600394    0.69994936
Proline                            -0.44059693  0.39335085    0.49811488
                           Flavanoids Nonflavanoid phenols Proanthocyanins
Class                      -0.8474975            0.4891092    -0.499129824
Alcohol                     0.2368149           -0.1559295     0.136697912
Malic acid                 -0.4110066            0.2929771    -0.220746187
Ash                         0.1150773            0.1862304     0.009651935
Alcalinity of ash          -0.3513699            0.3619217    -0.197326836
Magnesium                   0.1957838           -0.2562940     0.236440610
Total phenols               0.8645635           -0.4499353     0.612413084
Flavanoids                  1.0000000           -0.5378996     0.652691769
Nonflavanoid phenols       -0.5378996            1.0000000    -0.365845099
Proanthocyanins             0.6526918           -0.3658451     1.000000000
Color intensity            -0.1723794            0.1390570    -0.025249931
Hue                         0.5434786           -0.2626396     0.295544253
OD280/OD315 of diluted wines 0.7871939          -0.5032696     0.519067096
Proline                     0.4941931           -0.3113852     0.330416700
                           Color intensity         Hue
Class                         0.26566757 -0.61736921
Alcohol                       0.54636420 -0.07174720
Malic acid                    0.24898534 -0.56129569
Ash                           0.25888726 -0.07466689
Alcalinity of ash             0.01873198 -0.27395522
Magnesium                     0.19995001  0.05539820
Total phenols                -0.05513642  0.43368134
Flavanoids                   -0.17237940  0.54347857
Nonflavanoid phenols          0.13905701 -0.26263963
Proanthocyanins              -0.02524993  0.29554425
Color intensity               1.00000000 -0.52181319
Hue                          -0.52181319  1.00000000
OD280/OD315 of diluted wines -0.42881494  0.56546829
Proline                       0.31610011  0.23618345
                           OD280/OD315 of diluted wines    Proline
Class                                      -0.788229589 -0.6337168
Alcohol                                     0.072343187  0.6437200
Malic acid                                 -0.368710428 -0.1920106
Ash                                         0.003911231  0.2236263
Alcalinity of ash                          -0.276768549 -0.4405969
Magnesium                                   0.066003936  0.3933508
Total phenols                               0.699949365  0.4981149
Flavanoids                                  0.787193902  0.4941931
Nonflavanoid phenols                       -0.503269596 -0.3113852
Proanthocyanins                             0.519067096  0.3304167
```

```
Color intensity                      -0.428814942  0.3161001
Hue                                   0.565468293  0.2361834
OD280/OD315 of diluted wines          1.000000000  0.3127611
Proline                               0.312761075  1.0000000
```

So Nonflavanoid phenols and Proanthocyanins have highest correlation based on cor value 0.864563500095115; OD280/OD315 of diluted wines and Ash have thelowest correlation based on cor value 0.003911231.

3 What is the alcohol content of the wine with the highest color intensity?

```
data_wine[which(data_wine$Alcohol == max(data_wine$Alcohol)), ]["Color intensity"]
```

```
  Color intensity
9             5.2
```

4 What percentage of wines had a higher content of proanthocyanins compare to ash?

The percent is about 8.42%:

```
nrow(data_wine[which(data_wine$Proanthocyanins > data_wine$"Ash"), ])/nrow(data_wine)
```

```
[1] 0.08426966
```

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. (This table does not need to be "fancy" but should clearly identify what each value represents.)

```
apply(data_wine[c("Alcohol", "Malic acid", "Ash", "Alcalinity of ash", "Magnesium", "Total ph
```

```
                Alcohol                  Malic acid
            13.0006180                   2.3363483
                    Ash           Alcalinity of ash
              2.3665169                  19.4949438
              Magnesium              Total phenols
             99.7415730                   2.2951124
             Flavanoids        Nonflavanoid phenols
              2.0292697                   0.3618539
```

```
      Proanthocyanins              Color intensity
            1.5908989                    5.0580899
                  Hue OD280/OD315 of diluted wines
            0.9574494                    2.6116854
              Proline
          746.8932584
```

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. (You may use an existing R function to carry out the t-test, or for minor extra credit, manually write your own calculation of the t-test p-values.)

Use the R function: t.test(), and devide classes into 3 groups: 12, 23, 13

```
names(data_wine) = c("Class", "Alcohol", "Malic acid", "Ash", "Alcalinity of ash", "Magnesiu
dat12 = data_wine[which(data_wine$Class == 1 | data_wine$Class == 2),]
dat13 = data_wine[which(data_wine$Class == 1 | data_wine$Class == 3),]
dat23 = data_wine[which(data_wine$Class == 3 | data_wine$Class == 2),]
t.test(Tp~Class, dat12)
```

```
    Welch Two Sample t-test

data:  Tp by Class
t = 7.4206, df = 119.14, p-value = 1.889e-11
alternative hypothesis: true difference in means between group 1 and group 2 is not equal to
95 percent confidence interval:
 0.4261870 0.7364055
sample estimates:
mean in group 1 mean in group 2
       2.840169        2.258873
```

```
t.test(Tp~Class, dat13)
```

```
    Welch Two Sample t-test

data:  Tp by Class
t = 17.12, df = 98.356, p-value < 2.2e-16
alternative hypothesis: true difference in means between group 1 and group 3 is not equal to
95 percent confidence interval:
```

```
 1.026801 1.296038
sample estimates:
mean in group 1 mean in group 3
       2.840169        1.678750
```

```
t.test(Tp~Class, dat23)
```

```
    Welch Two Sample t-test

data:  Tp by Class
t = 7.0125, df = 116.91, p-value = 1.622e-10
alternative hypothesis: true difference in means between group 2 and group 3 is not equal to
95 percent confidence interval:
 0.4162855 0.7439610
sample estimates:
mean in group 2 mean in group 3
       2.258873        1.678750
```

All of p-values are smaller than 0.05, which means differences of the level of phenols across from three classes are big.

Additionally, differences between means are all in the 95 percent confidence interval. It means that values of each groups are same with the whole area.

And use R to build a function to compute t value, but didn't find the way to compute p value:

```
myT <-function(val1, val2, data){
  # val1 is the target var to compute t value
  # val2 is the name of classes, which is a binary var in data
  se1 = data[which(data[val2] == data[val2][[1]][1]),][val1]
  se2 = data[which(data[val2] != data[val2][[1]][1]),][val1]
  t=(lapply(se1, mean)[[1]]-lapply(se2, mean)[[1]])/sqrt(lapply(se1,sd)[[1]]**2/length(se1)+
  return(list("t value"=t))
}

myT("Tp", "Class", dat12)
```

```
$`t value`
[1] 0.9052817
```

Problem 2 - AskAManager.org Data Please download this dataset. It is from an ongoing salary survey from AskAManager.org. We're going to do some data cleaning to prepare it for an analysis.

a: 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.

```
data_aam = read.csv("./AskAManager.csv", head = 1)
```

b: Clean up the variable names. Simplify them.

Rename them and print renamed names:

```
names(data_aam)<-c("X", "Time", "Age", "Industry", "Job", "Job_more", "Salary", "ExtraSalary"
names(data_aam)
```

```
 [1] "X"             "Time"          "Age"           "Industry"
 [5] "Job"           "Job_more"      "Salary"        "ExtraSalary"
 [9] "Currency"      "CurrencyO"     "IncomeContext" "Country"
[13] "State"         "City"          "OverallYear"   "FieldYear"
[17] "Education"     "Gender"        "Race"
```

c: 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.

Use which to choose items that have value "USD" in "Currency", and print the number of rows of original data frame and of filtered data frame.

```
data_aam_us = data_aam[which(data_aam$Currency == "USD"),]
print(nrow(data_aam))
```

```
[1] 28062
```

```
print(nrow(data_aam_us))
```

```
[1] 23374
```

d: 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. (Hint: Making these variables factor may make your life easier.)

Because age given is a range of character format, use "dic_age" which likes dictionary in python to fastly invert age character to integer. Use a "dic_year" for working years too in a similar way.

Secondly, filter them and print the number of rows.

```
dic_age = c("18-24"=24, "25-34"=34, "35-44"=44, "45-54"=54, "55-64"=64, "65 or over"=1000)
dic_years = c("5-7 years" = 5, "8 - 10 years" = 8, "2 - 4 years" = 2, "21 - 30 years" = 21, "
data_aam_age = data_aam[which(dic_age[data_aam$Age] > 17+dic_years[data_aam$OverallYear]),]
data_aam_age = data_aam_age[which(dic_age[data_aam_age$Age] > 17+dic_years[data_aam_age$Fiel
print(nrow(data_aam))
```

```
[1] 28062
```

```
print(nrow(data_aam_age))
```

```
[1] 27990
```

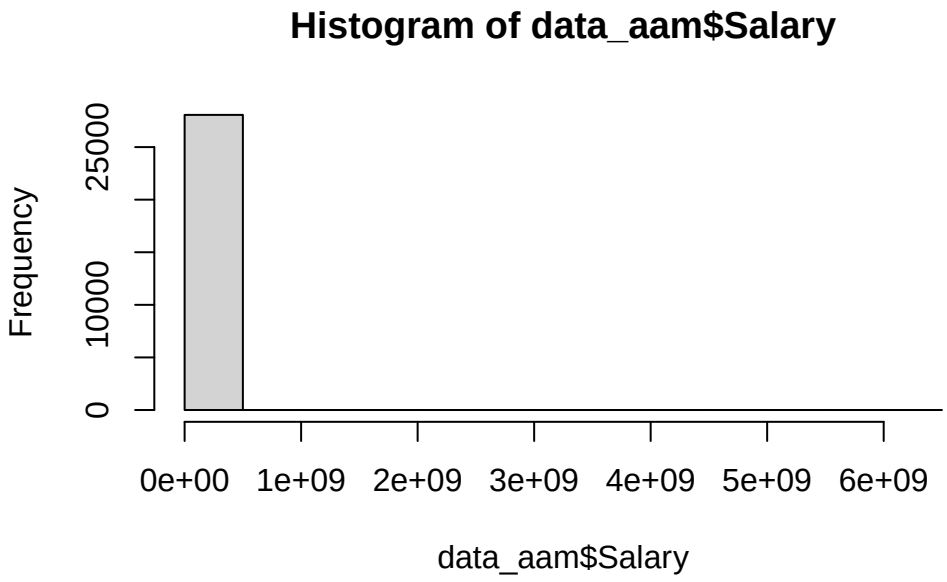
e: A lot of the incomes are likely false. Eliminate any rows with extremely low or extremely high salaries. I'll leave the decision of what thresholds to use up to you; you could choose to eliminate only impossible values, or you could restrict the sample to eliminate the extreme values even if they are realistic (e.g. removing the billionaires or the folks making $<$ $1,000 per year). You must justify your choice, along with either a cited source or an exploration the data, or some combination.Report your final sample size.

First, I think it is necessary to analyse the data. Base on the summary and graph, here are some etremely big values which I need to delete.

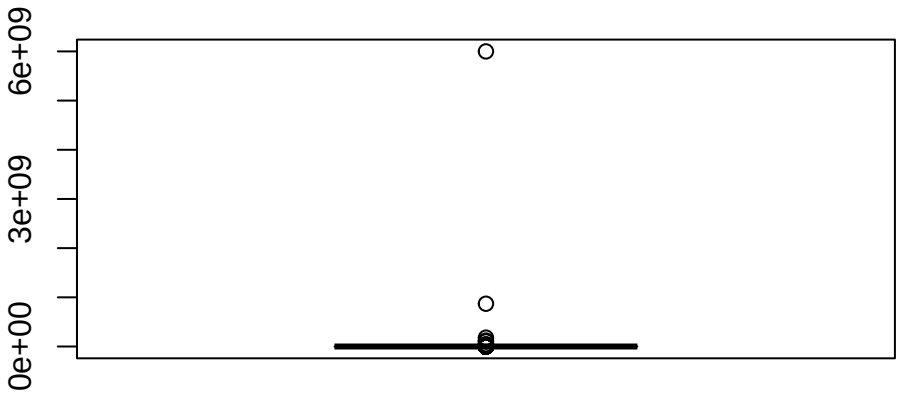
```
summary(data_aam$Salary)
```

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
0.000e+00 5.400e+04 7.500e+04 3.614e+05 1.100e+05 6.000e+09
```

```
hist(data_aam$Salary)
```




```
boxplot(data_aam$Salary)
```

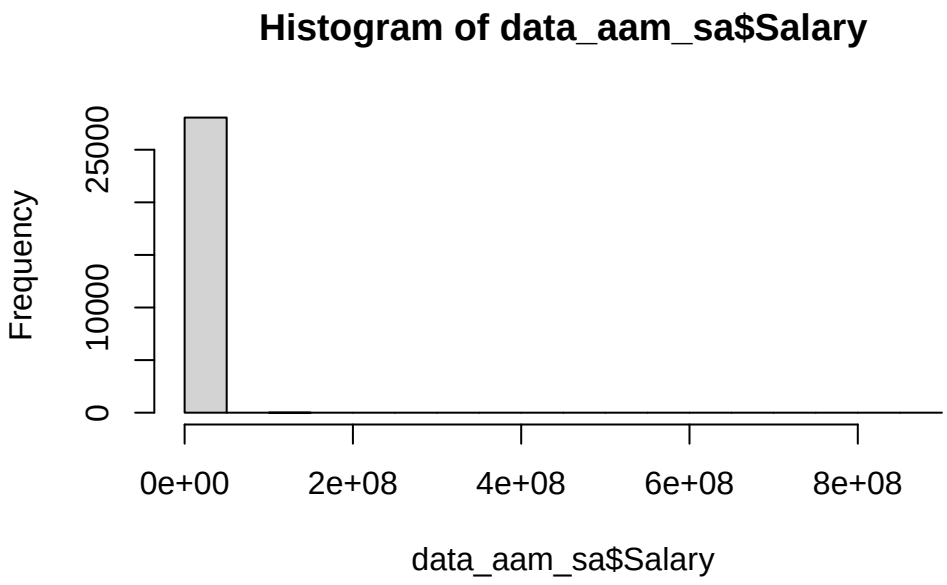


After deleting rows with salary higher than 1e+9, analyse the data again. Because here are still lots of extremely values, choose to delete rows with salary higher than 1e+7 based on "3rd Qu" value. Finally, max value is 8800000 and 3rd Qu value is 109575, and it seems more reliable.

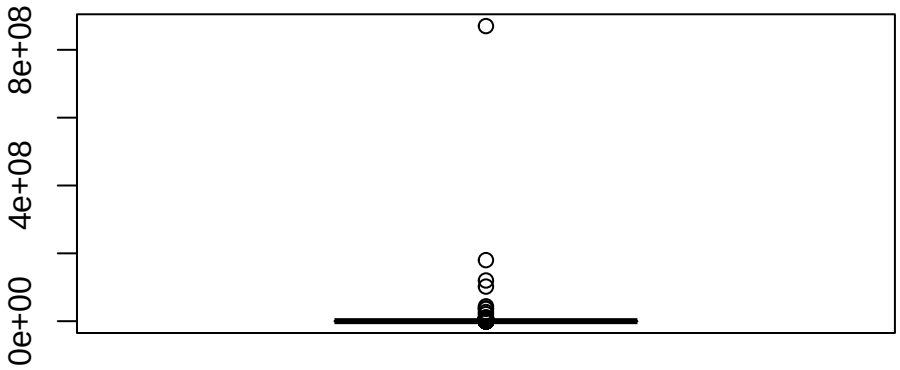
```
data_aam_sa=data_aam[which(data_aam$Salary < 1e+9),]
summary(data_aam_sa$Salary)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      0   54000   75000  147615  110000 870000000
```

```r
hist(data_aam_sa$Salary)
```



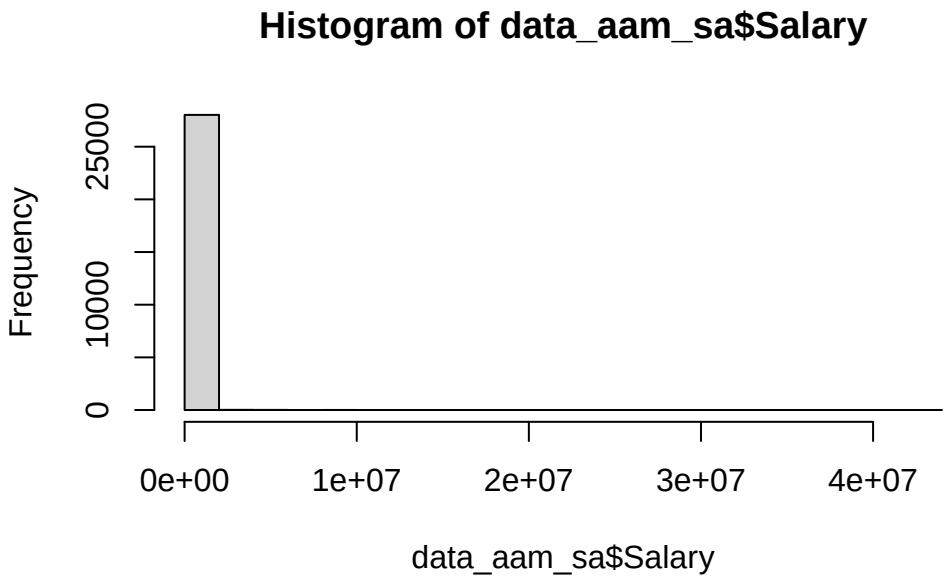

```r
boxplot(data_aam_sa$Salary)
```



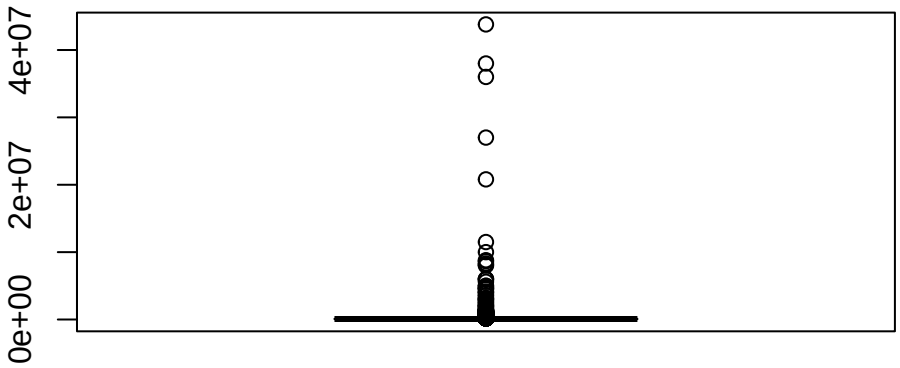

```r
data_aam_sa = data_aam[which(data_aam$Salary < 1e+8),]
summary(data_aam_sa$Salary)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
       0    54000    75000   102300   109769 43800000
```

```r
hist(data_aam_sa$Salary)
```




```
boxplot(data_aam_sa$Salary)
```



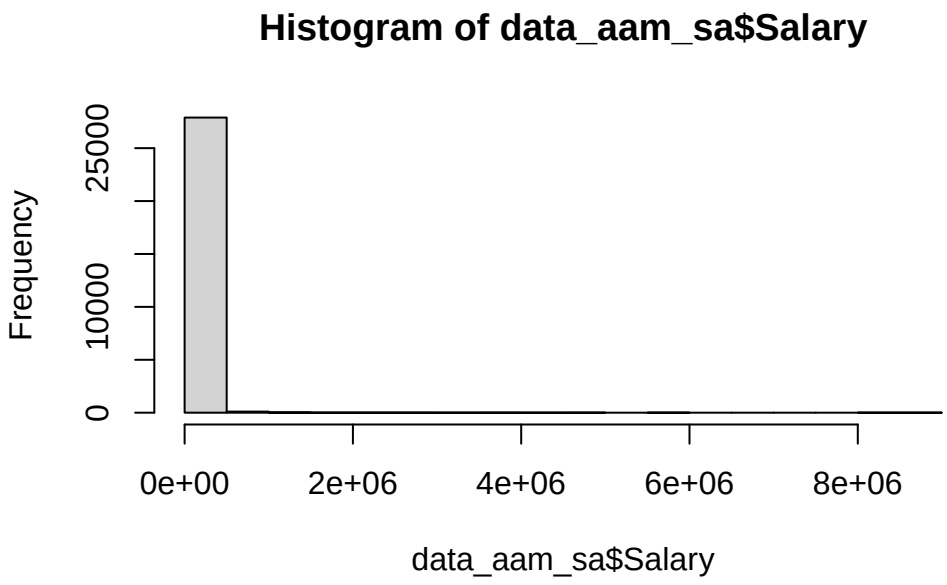

```
data_aam_sa = data_aam[which(data_aam$Salary < 1e+7),]
nrow(data_aam_sa)
```

```
[1] 28050
```

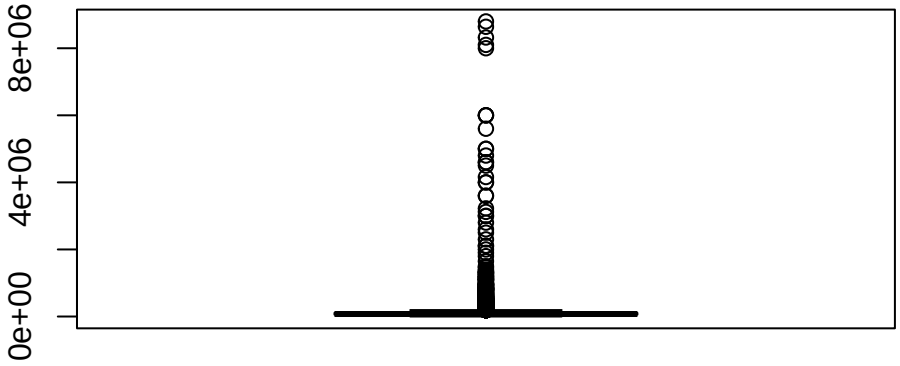
```
summary(data_aam_sa$Salary)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
      0   54000   75000   95655  109575 8800000 
```

```r
hist(data_aam_sa$Salary)
```




```r
boxplot(data_aam_sa$Salary)
```



About the extremely small values, based on the sorted salary list, values less than 100 are most likely to be false. Moreover, values of salary which is less than 2000 are also difficult to understand.

Here are codes, but output is too big, so they are annotations.

```r
# sort(data_aam_sa$Salary)
# table(data_aam_sa$Salary)
```

So finally delete rows with salary bigger than 1e+7 and smaller than 2000, size is 27932.

```
data_aam_sa = data_aam[which(data_aam$Salary > 2000 & data_aam$Salary < 1e+7),]
nrow(data_aam_sa)
```

```
[1] 27932
```

Problem 3 - Palindromic Numbers Palindromic numbers are integers that are equal to the reverse of their digits. For example, 59195 is palindromic, whereas 59159 is not.

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 (see instructions above).

Here used c() and rev function to finish it:

```
isPalindromic <- function(inter){
  an = inter
  li_nu = c()
  lis = c()
  # save int as a list
  while(an > 0){
    li_nu = append(li_nu, an %% 10)
    an = (an-(an%%10)) / 10
  }
  # try reverse
  if(all(rev(li_nu) == li_nu)){
    lis = list("isPalindromic"=TRUE, "reversed"= inter)
  }
  else{
    ans = 0
    # return reverse number if not palindromic
    for(i in 0:length(li_nu)){
      ans = ans + li_nu[length(li_nu)-i]*(10**i)
      if(i == length(li_nu)-1){
        lis = list("isPalindromic"=FALSE, "reversed" =ans)
      }
    }

  }
  return(lis)
}

isPalindromic(212)
```

```
$isPalindromic
[1] TRUE

$reversed
[1] 212
```

```
isPalindromic(59159)
```

```
$isPalindromic
[1] FALSE

$reversed
[1] 95195
```

```
isPalindromic(59195)
```

```
$isPalindromic
[1] TRUE

$reversed
[1] 59195
```

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.

```
nextPalindrome <- function(inter){
  if(isPalindromic(inter)$isPalindromic){
    return(inter)
  }
  while(1){
    inter = inter + 1
    if(isPalindromic(inter)$isPalindromic){
      return(inter)
    }
  }
}
```

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

i. 391

```
nextPalindrome(391)
```

```
[1] 393
```

ii. 9928

```
nextPalindrome(9928)
```

```
[1] 9999
```

iii. 19272719

```
nextPalindrome(19272719)
```

```
[1] 19277291
```

iv. 109

```
nextPalindrome(109)
```

```
[1] 111
```

v. 2

```
nextPalindrome(2)
```

```
[1] 2
```