Data Mining Using R

Karthick Sharan

2/7/2022

R Markdown

Problem 1

```
# Problem 1(a)
x <- c(1, 2.3, 2, 3, 4, 8, 12, 43, -4, -1)
x
```

```
##  [1]  1.0  2.3  2.0  3.0  4.0  8.0 12.0 43.0 -4.0 -1.0
```

```
# the function c() is used to creates a vector of arguments mentioned above and assign it
#to the varibale 'x'
```

```
# Problem 1(b)
max(x)
```

```
## [1] 43
```

```
# the above function max() is used to return the maximum value from the given arguments
```

```
# Problem 1(c)
y <- c(x,NA)
y
```

```
##  [1]  1.0  2.3  2.0  3.0  4.0  8.0 12.0 43.0 -4.0 -1.0   NA
```

```
# appends 'NA' (NULL value) to end of x
```

```
# Problem 1(d)
max(y, na.rm = T)
```

```
## [1] 43
```

```r
# returns max of y after removing NULL values from the passed arguments
```

```r
# Problem 1(e)
x2 <- c(-100, -43, 0, 3, 1, -3)
min(x,x2)
```

```
## [1] -100
```

```r
# returns the element with the minimum value within the union of x and x2
```

```r
# Problem 1(f)
sample(4:10)
```

```
## [1]  6  8 10  7  5  9  4
```

```r
# returms a random sample from 4:10 of length(4:10) elements without replacement
```

```r
# Problem 1(g)
sample(c(2,5,3), size=3, replace=FALSE)
```

```
## [1] 5 3 2
```

```r
# returns a random sample from vector c(2,3,5) of 3 elements without replacement
```

```r
# Problem 1(h)
sample(c(2,5,3), size=3, replace= TRUE)
```

```
## [1] 2 3 3
```

```r
# returns a random sample from vector c(2,3,5) of 3 elements with replacement (bootstrap sampling).
```

```r
# Problem 1(i)
sample(2, 10, replace = TRUE) # (what happens when you write "replace = FALSE"?)
```

```
##  [1] 1 1 2 1 1 1 1 2 1 1
```

```r
# for replace=TRUE, function returns a random sample of size 10, with elements from 1:2
# if replace+FALSE, the functions throws an error since 1:2 is only 2 elements but
# selection size is given as 10, sample size cannot be greater than the population size.
```

```r
# Problem 1(j)
sample(1:2, size=10, prob=c(1,3), replace=TRUE)
```

```
##  [1] 2 2 2 1 2 2 2 2 2 2
```

```r
# returns a sample of size 10 from population = 1:2, where probablity of 2 getting chosen
# is 3/4 and probablity of 1 getting picked is 1/4. So it's not random sampling but rather
# is more like a weighted bootstrap sampling.
```

```r
# Problem 1(k)
round(3.14159, digits = 2)
```

```
## [1] 3.14
```

```r
# returns the value of 3.14159 with precision of 2 decimal points, hence returns 3.14
```

```r
# Problem 1(l)
range(100,400)
```

```
## [1] 100 400
```

```r
# returns the minimum and maximum value from the given arguments. In this case returns a vector
# of 100 and 400
```

```r
# Problem 1(m)
matrix(c(1, 2.3, 2, 3, 4, 8, 12, 43, -4, -1, 9, 14), nr=3, nc=4)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]  1.0    3   12   -1
## [2,]  2.3    4   43    9
## [3,]  2.0    8   -4   14
```

```r
# creates a matrix from the arguments with 3 rows and 4 columns (values for nr and nc)
```

```r
# Problem 1(n)
matrix(c(1, 2.3, 2, 3, 4, 8, 12, 43, -4, -1, 9, 14), nr=3, nc=4, byrow = T)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1  2.3    2    3
## [2,]    4  8.0   12   43
## [3,]   -4 -1.0    9   14
```

```r
# creates a marix of 3x4 dimension, the elements in the matrix are filled by rows instead of columns.
```

```r
# Problem 1(o)
x <- matrix(c(4,3,4,6,7,6),3,2)
rownames(x) <- c("row1","row2","row3")
colnames(x) <- c("col1", "col2")
x
```

```
##      col1 col2
## row1    4    6
## row2    3    7
## row3    4    6
```

```r
# creates a 3x2 matrix from the given arguments and stores it in variable x.
# the rownames() and columnnames() functions are used to name the rows and column of matrix x
```

```r
# Problem 1(p)
x <- rbind(c(1:4),c(5,8))
print(x)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    5    8    5    8
```

```r
# creates a matrix x by stacking the 2 argument vectors vertically (both vectors are different rows)
y <- cbind(c(1:4),c(5,8))
print(y)
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    2    8
## [3,]    3    5
## [4,]    4    8
```

```r
# creates a matrix x by stacking the 2 argument vectors horizontaly (both vectors are different columns
```

```r
# Problem 1(q)
y <- 1:9
w <- 2:10
z <- 3:5
rbind(y,w,z)
```

```
##   [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
## y    1    2    3    4    5    6    7    8    9
## w    2    3    4    5    6    7    8    9   10
## z    3    4    5    3    4    5    3    4    5
```

```r
# creates a matrix where x,w and z are all seperate rows. Vertically stacking x,w and z.
```

```r
# Problem 1(r)
m <- matrix(1:36,9,4)
m[2,3]
```

```
## [1] 20
```

```r
# returns the (2,3) element, from matrix m (element in 2nd row, 3rd column)
m[,3]
```

```
## [1] 19 20 21 22 23 24 25 26 27
```

```
# retunrs all rows of 3rd column from matrix m
m[2,]
```

```
## [1]  2 11 20 29
```

```
# returns all elemnts from 2nd row (2nd row all columns)
cbind(m[,3])
```

```
##       [,1]
##  [1,]   19
##  [2,]   20
##  [3,]   21
##  [4,]   22
##  [5,]   23
##  [6,]   24
##  [7,]   25
##  [8,]   26
##  [9,]   27
```

```
# binnds the 3rd column from matrix m into a seperate column (creates a single column matrix
# with elements from 3rd column of matrix m)
m[,-3]
```

```
##       [,1] [,2] [,3]
##  [1,]    1   10   28
##  [2,]    2   11   29
##  [3,]    3   12   30
##  [4,]    4   13   31
##  [5,]    5   14   32
##  [6,]    6   15   33
##  [7,]    7   16   34
##  [8,]    8   17   35
##  [9,]    9   18   36
```

```
# removes the 3rd column form matrix m, resulting matrix will be of dimension 9x3
m[-(3 : 8),2:4]
```

```
##      [,1] [,2] [,3]
## [1,]   10   19   28
## [2,]   11   20   29
## [3,]   18   27   36
```

```
# removes the rows from 3rd to 8th and selects columns from 2nd to 4th from matrix m
# resulting matrix will be 3x3 with 1st,2nd and 9th row and 2nd, 3rd and 4th columns.
```

```
# Problem 1(s)
x <- cbind(x1 = 3, x2 = c(4:1, 2:5))
# binds together both the arguments as columns and creates a matrix
dimnames(x)[[1]] <- letters[1:8]
# renames the rownames to fisrt eight alphabets, (if we use [[2]] we can rename columns)
apply(x, 2, mean, trim=.2)
```

```
## x1 x2
##  3  3
```

```
# retuns the mean of both columns, column wise mean hence a vector with 2 elements would be retunred
col.sums <- apply(x, 2, sum)
# col.sums is assigned a vector of 2 elements each representing the sum of the respective column
row.sums <- apply(x, 1, sum)
# row-wise sum is calculated and assigned to row.sums. (rowsum is a vector of 8 elements)
apply(x, 2, sort)
```

```
##      x1 x2
## [1,]  3  1
## [2,]  3  2
## [3,]  3  2
## [4,]  3  3
## [5,]  3  3
## [6,]  3  4
## [7,]  3  4
## [8,]  3  5
```

```
# the matrix is sorted in ascending order colum-wise.
```

Problem 2

```
## Problem 2(a)
x <- 15
y <- c(1,2,3,10,100)
z <- x*y
sum(z)
```

```
## [1] 1740
```

```
# Problem 2(b)
s1 <- 0:10   # alternativeky we can also use sqe() function
s2 <- -5:5
print(s1)
```

```
##  [1]  0  1  2  3  4  5  6  7  8  9 10
```

```
print(s2)
```

```
##  [1] -5 -4 -3 -2 -1  0  1  2  3  4  5
```

```
# Problem 2(c)
s3 <- seq(-3,3,0.1)
s3
```

```
##  [1] -3.0 -2.9 -2.8 -2.7 -2.6 -2.5 -2.4 -2.3 -2.2 -2.1 -2.0 -1.9 -1.8 -1.7 -1.6
## [16] -1.5 -1.4 -1.3 -1.2 -1.1 -1.0 -0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1
## [31]  0.0  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1.0  1.1  1.2  1.3  1.4
## [46]  1.5  1.6  1.7  1.8  1.9  2.0  2.1  2.2  2.3  2.4  2.5  2.6  2.7  2.8  2.9
## [61]  3.0
```

```
# Problem 2(d)
t <- c('mon','tue','wed','thu','fri','sat')
m <- c(90,80,50,20,5,20)
study <- cbind(t,m)  # we can use data.fram if we don't want implicit type conversion
study
```

```
##      t     m
## [1,] "mon" "90"
## [2,] "tue" "80"
## [3,] "wed" "50"
## [4,] "thu" "20"
## [5,] "fri" "5"
## [6,] "sat" "20"
```

```
# Probelm 2(e)
df <- data.frame(age = c(21,35,829,2), sex = c('m','f','m','e'), height = c(181,173,171,166),
                 weigth =c(69,58,75,60))
print(min(df$age))
```

```
## [1] 2
```

```
print(max(df$age))
```

```
## [1] 829
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
to_remove <- df[,1]<20 | df[,1]>80  # to_remove is a boolean mask
df[,1][to_remove] <- NA
df$BMI <- df$weigth / df$height  # calculate BMI and apend it to df
df
```

```
##   age sex height weigth       BMI
## 1  21   m    181     69 0.3812155
## 2  35   f    173     58 0.3352601
## 3  NA   m    171     75 0.4385965
## 4  NA   e    166     60 0.3614458
```

Problem 3

```r
## Problem 3(a)
x <- c(9, 8, 12, 6, 1, 10, 10, 10, 8, 516, 8, 6, 4, 19, 100)
mean(x)
```

```
## [1] 48.46667
```

```r
# Problem 3(b)
sd(x)
```

```
## [1] 131.5261
```

```r
# Problem 3(c)
range(x)
```

```
## [1]   1 516
```

```r
# Problem 3(d)
fivenum(x)
```

```
## [1]   1   7   9  11 516
```

```r
# Problem 3(e)
is.null(x)
```

```
## [1] FALSE
```

```r
# Problem 3(f)
# just from observing we can see that there are multiple outliers in data
q1 <- quantile(x,0.25)
q3 <- quantile(x,0.75)
iqr = q3-q1
x_no_out <- x[x>q1-1.5*iqr & x<q3+1.5*iqr]
x_no_out
```

```
##  [1]  9  8 12  6 10 10 10  8  8  6  4
```

Problem 4

```r
## Problem 4(a)
df <- read.csv("C:/Masters - Business Analytics/Data Mining/assignment 1/arbuthnot.csv")
dim(df)
```

```
## [1] 82  4
```

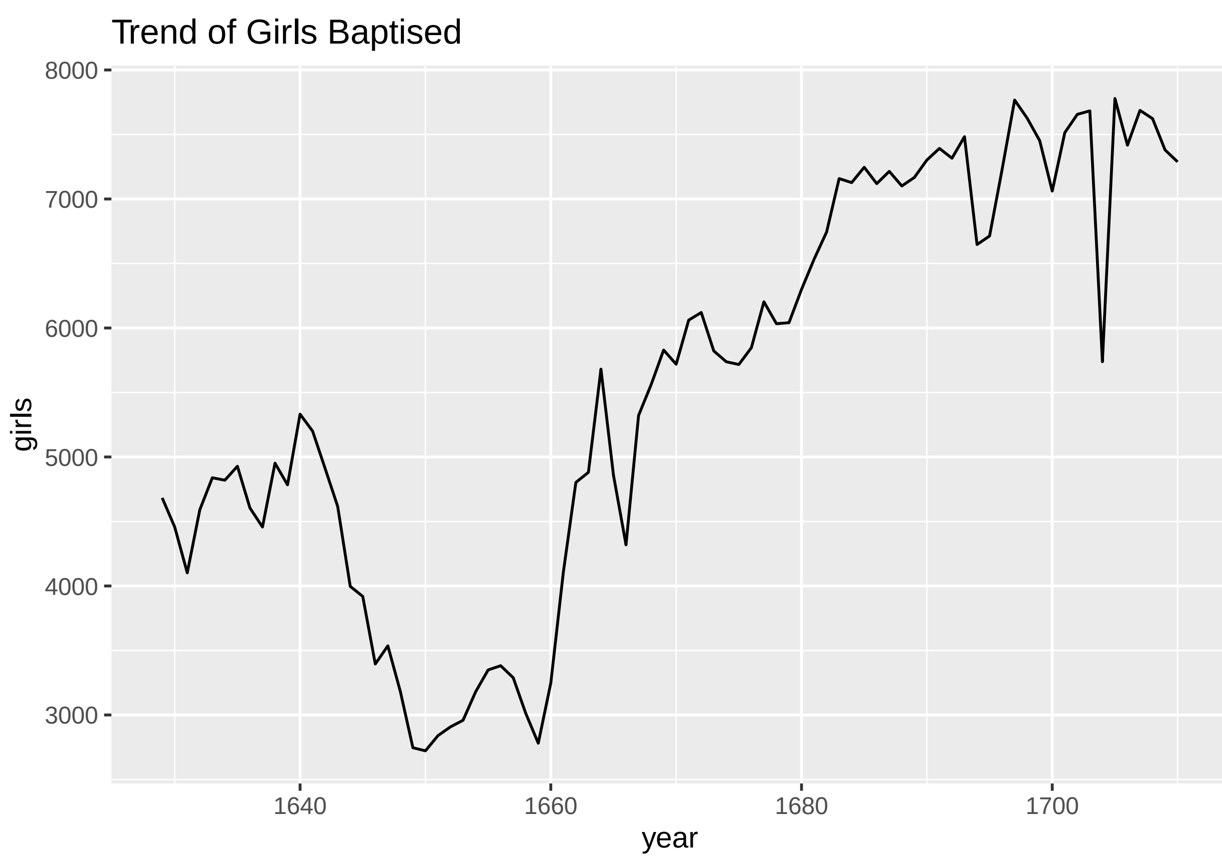
```r
# Problem 4(b)
names(df)
```

```
## [1] "X"     "year"  "boys"  "girls"
```

```r
# Problem 4(c)
sum(df["girls"])
```

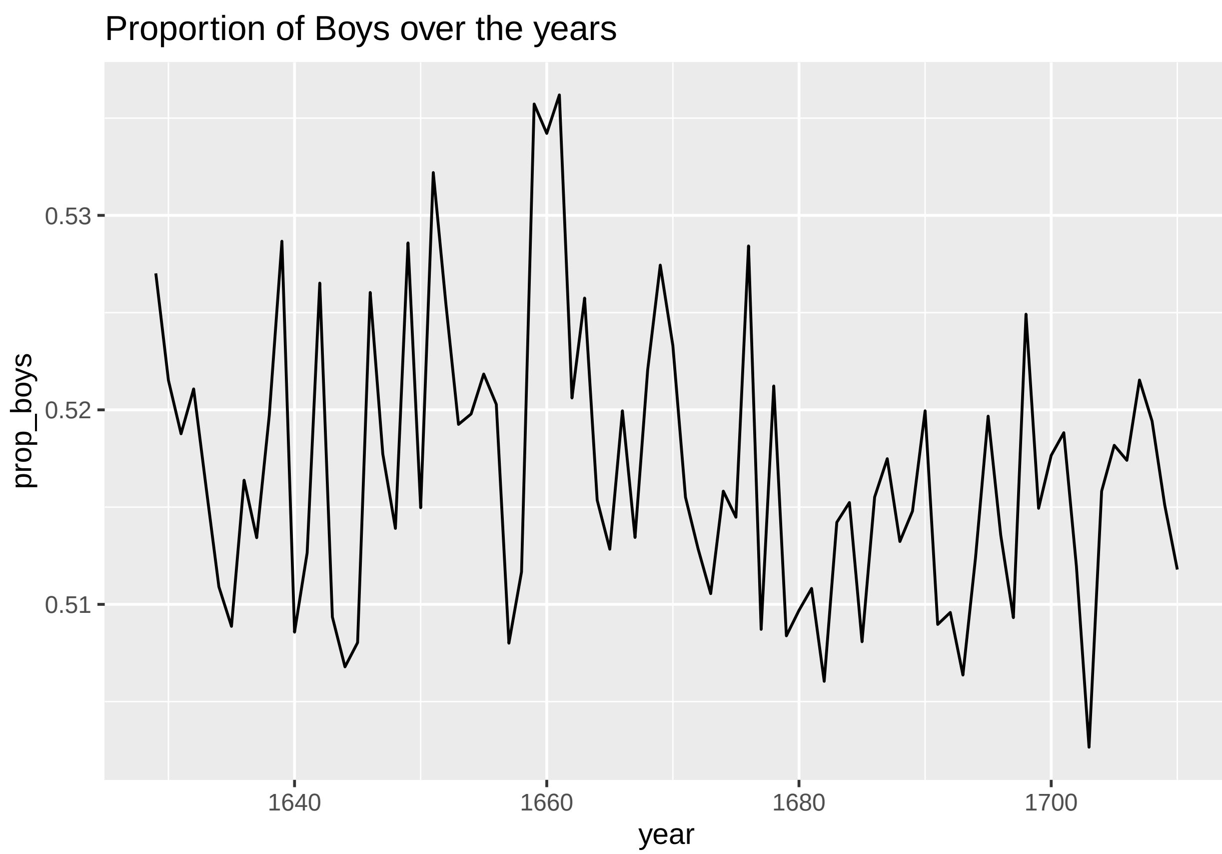
```
## [1] 453841
```

```r
# Problem 4(d)
ggplot(df, aes(x=year, y=girls)) + geom_line() +ggtitle("Trend of Girls Baptised")
```




```r
# girls baptised started declining initially unitl 1660, and then there is sharp increase
# till 1700 where it gets saturated.
```

```r
# Problem 4(e)
library(tidyverse)
df <- mutate(df, prop_boys = boys/(boys+girls))
ggplot(df, aes(x=year, y=prop_boys))+geom_line()+ggtitle("Proportion of Boys over the years")
```




```
# There is no apparent trend, there were some very minor fluctuations but the proportion
# always remains close to 50%
```

```
# Problem 4(f)
mutate(df, births = boys+girls) %>%
arrange(desc(births)) %>%
  head(1)
```

```
##    X year boys girls prop_boys births
## 1 77 1705 8366  7779  0.518179  16145
```

Problem 5

```
## Problem 5(a)
data("attitude")
df <- attitude
summary(df)
```

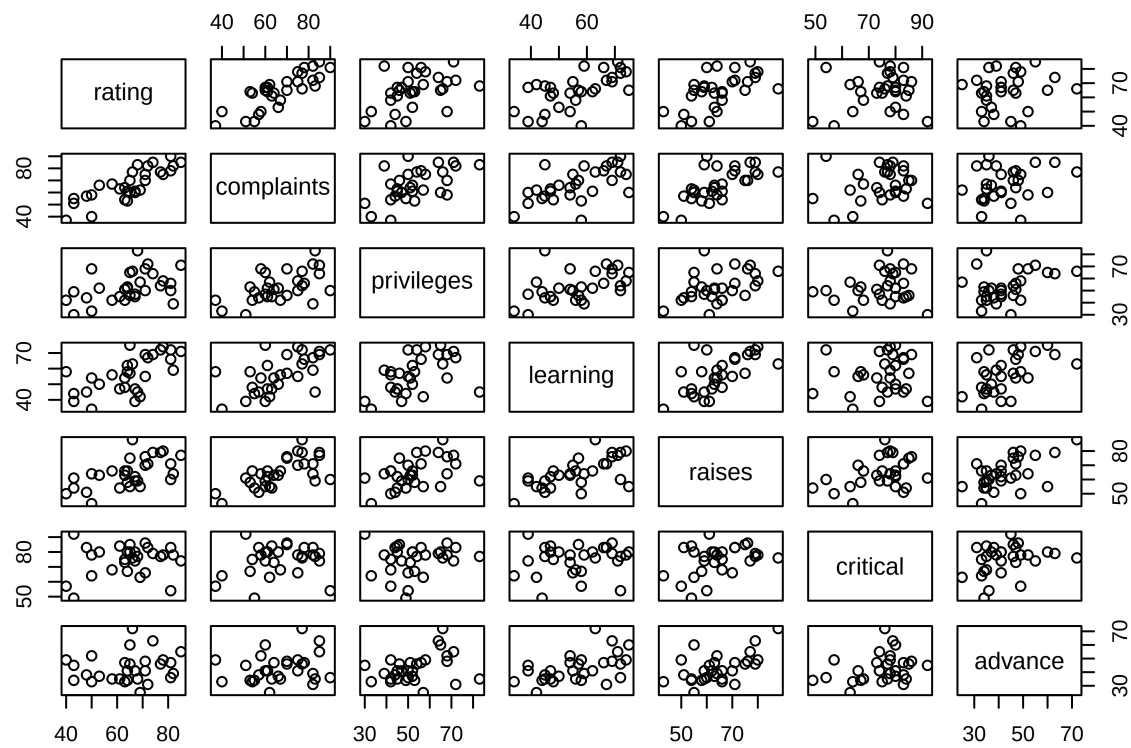
```
##      rating        complaints     privileges      learning         raises
##  Min.   :40.00   Min.   :37.0   Min.   :30.00   Min.   :34.00   Min.   :43.00
##  1st Qu.:58.75   1st Qu.:58.5   1st Qu.:45.00   1st Qu.:47.00   1st Qu.:58.25
##  Median :65.50   Median :65.0   Median :51.50   Median :56.50   Median :63.50
##  Mean   :64.63   Mean   :66.6   Mean   :53.13   Mean   :56.37   Mean   :64.63
```

```
##  3rd Qu.:71.75   3rd Qu.:77.0   3rd Qu.:62.50   3rd Qu.:66.75   3rd Qu.:71.00
##  Max.   :85.00   Max.   :90.0   Max.   :83.00   Max.   :75.00   Max.   :88.00
##     critical        advance
##  Min.   :49.00   Min.   :25.00
##  1st Qu.:69.25   1st Qu.:35.00
##  Median :77.50   Median :41.00
##  Mean   :74.77   Mean   :42.93
##  3rd Qu.:80.00   3rd Qu.:47.75
##  Max.   :92.00   Max.   :72.00
```

```r
# Problem 5(b)
nrow(df)*ncol(df) #also can use: count(df)*length(df)
```

```
## [1] 210
```

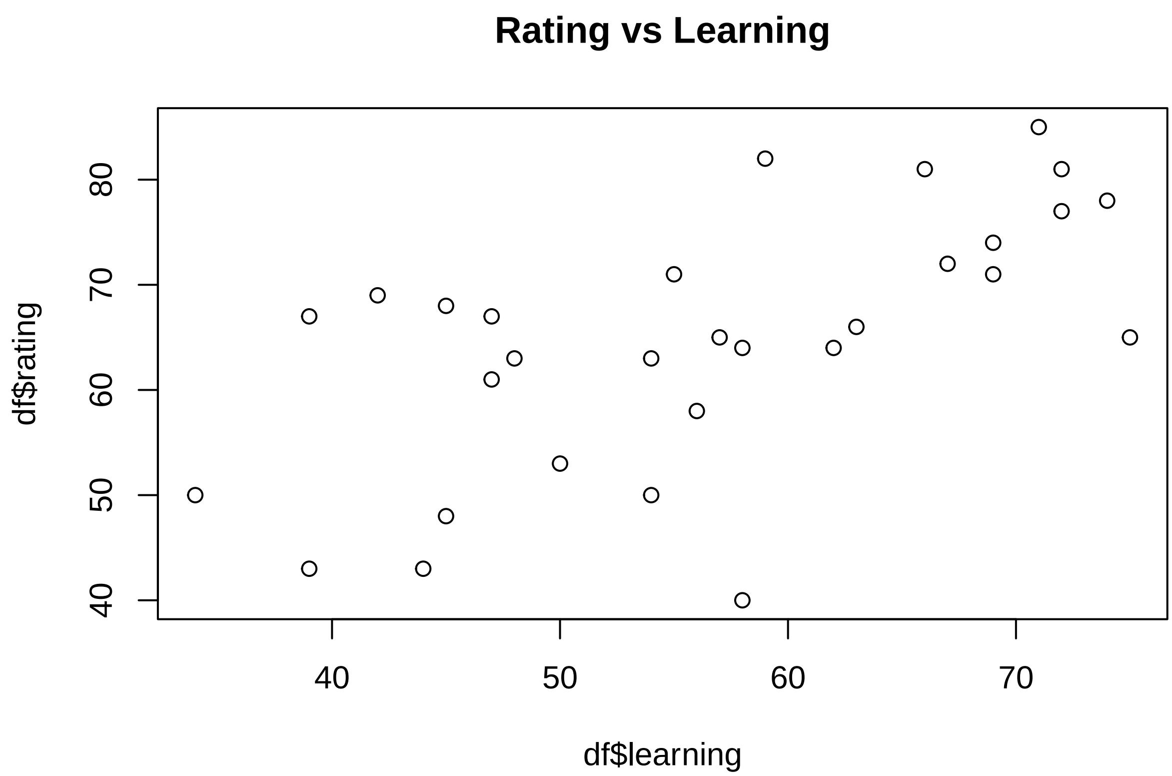
```r
# Problem 5(c)
plot(df)
```



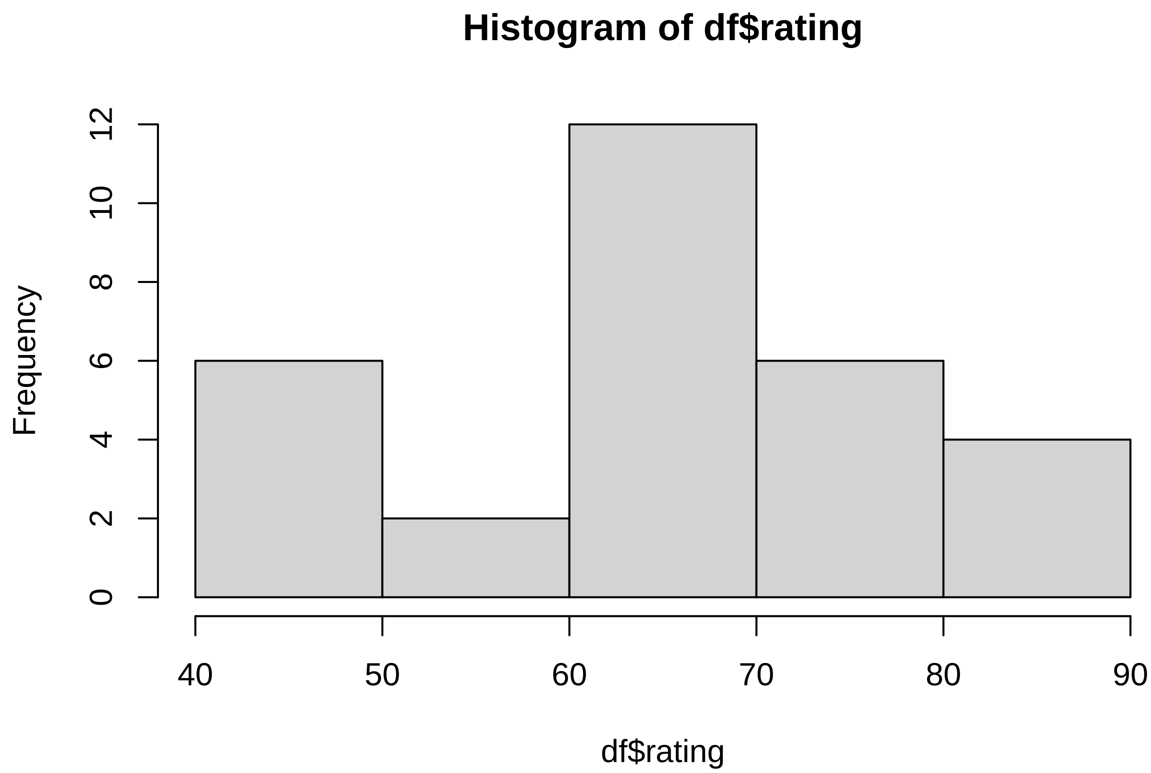

```r
# we can observe that "complaints" is the most correlated with "rating"
cor(df$rating, df$complaints) # to find the correlation value
```

```
## [1] 0.8254176
```

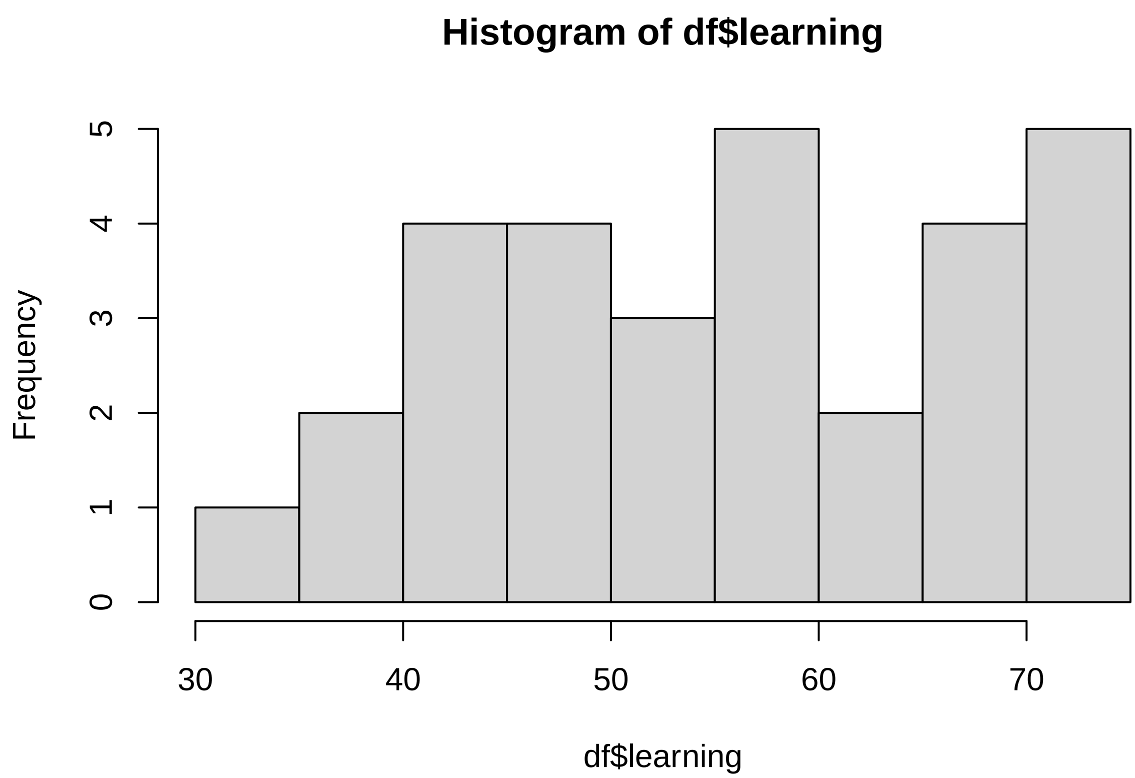
```r
# Problem 5(d)
plot(df$learning,df$rating,main = "Rating vs Learning")
```




```r
# Problem 5(e)
hist(df$rating)
```




```
hist(df$learning)
```




```
par(mfrow=c(2,2))
```

Problem 6

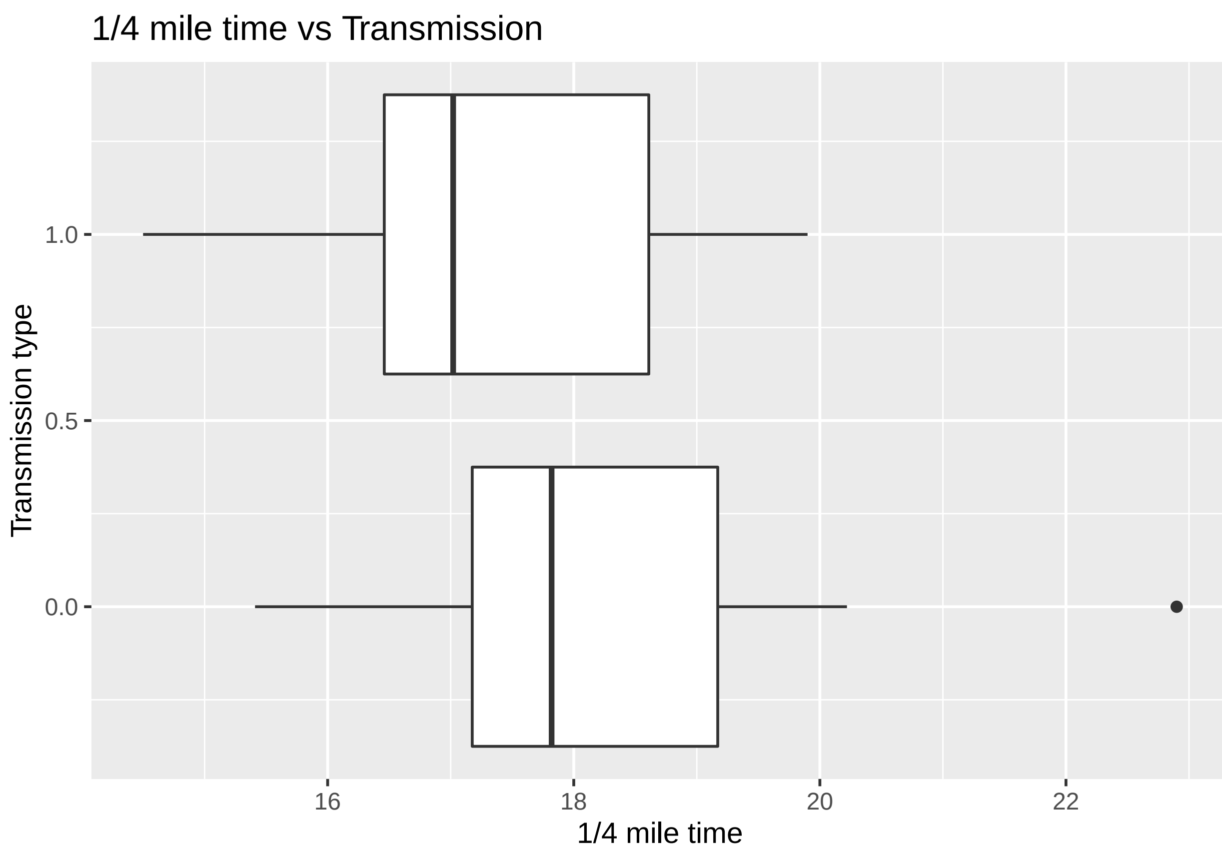
```
## Problem 6(a)
data("mtcars")
df <- mtcars
df
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
```

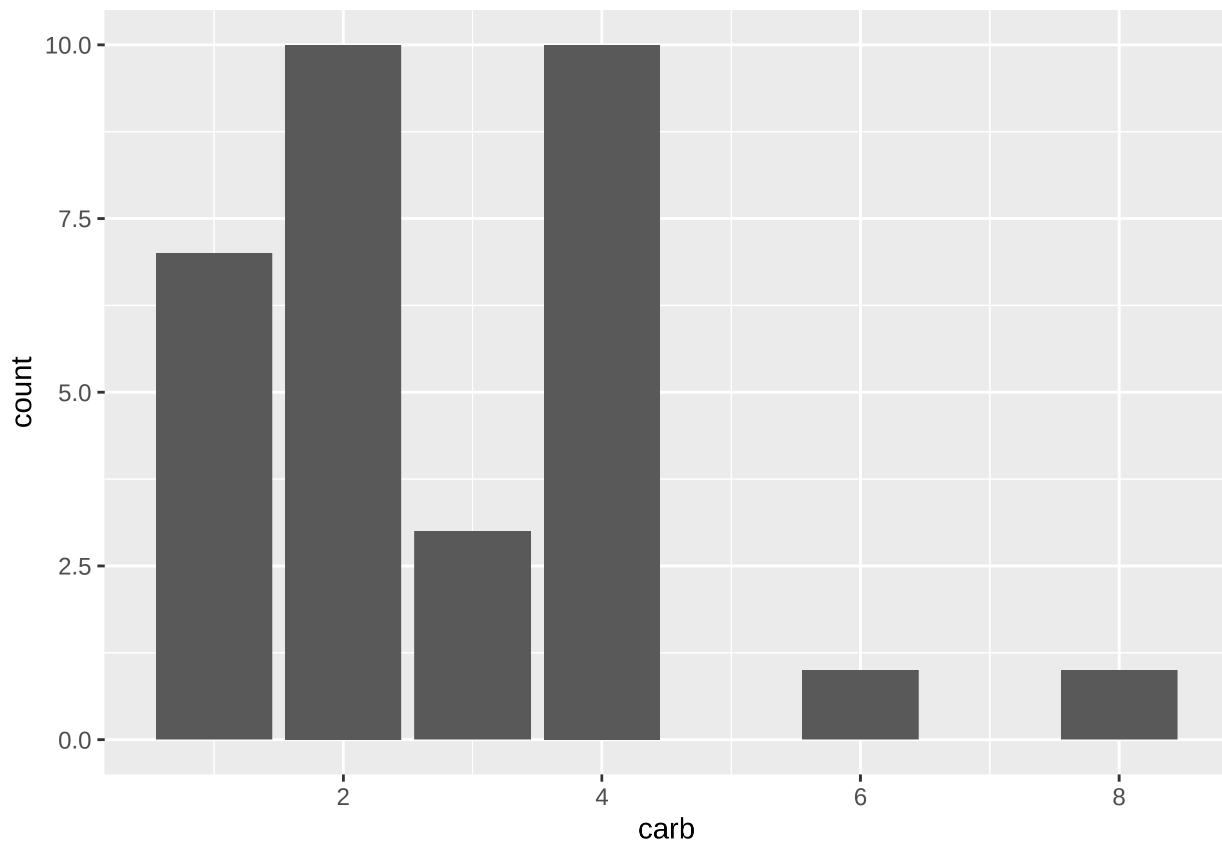
```
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```r
# The dataset "mtcars" comprises of fuel consumption and 10 aspects of automobile design
# and performance for 32 automobiles.
# It has a lot of useful variables such as mpg, Number of cylinder, transmission type, weight etc.
```

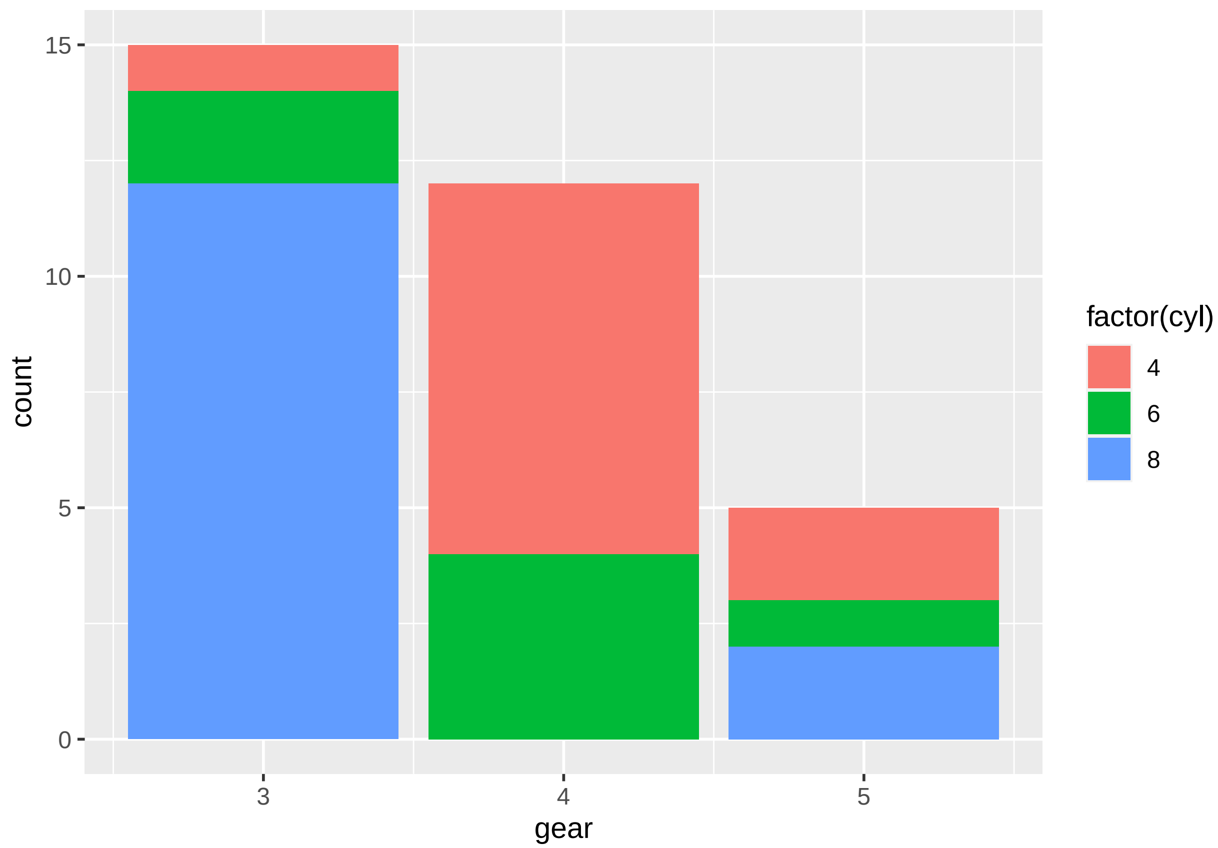
```r
# Problem 6(b)
ggplot(df, aes(x=qsec, y=am, group=factor(am)))+geom_boxplot()+
  ggtitle("1/4 mile time vs Transmission")+xlab("1/4 mile time")+
  ylab("Transmission type")
```



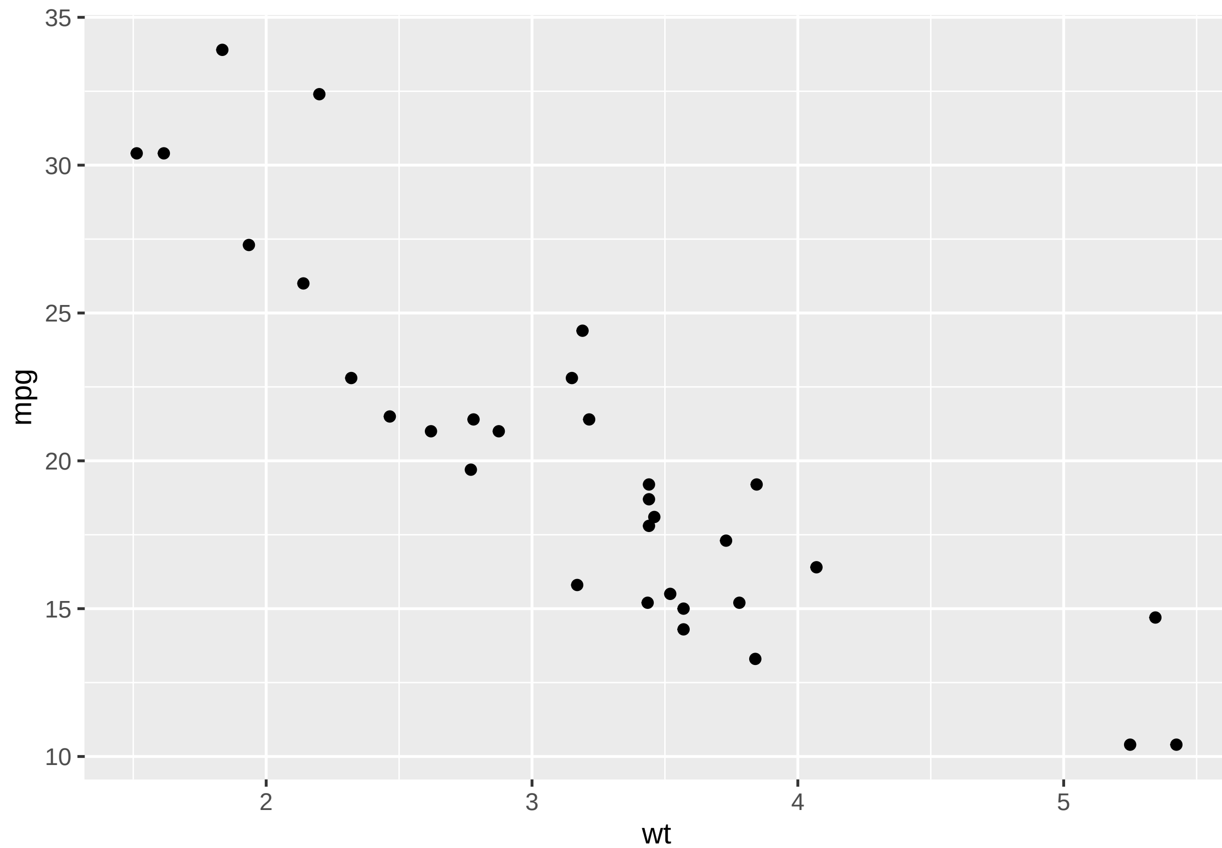

```r
# Problem 6(c)
ggplot(df, aes(x=carb))+geom_bar()
```



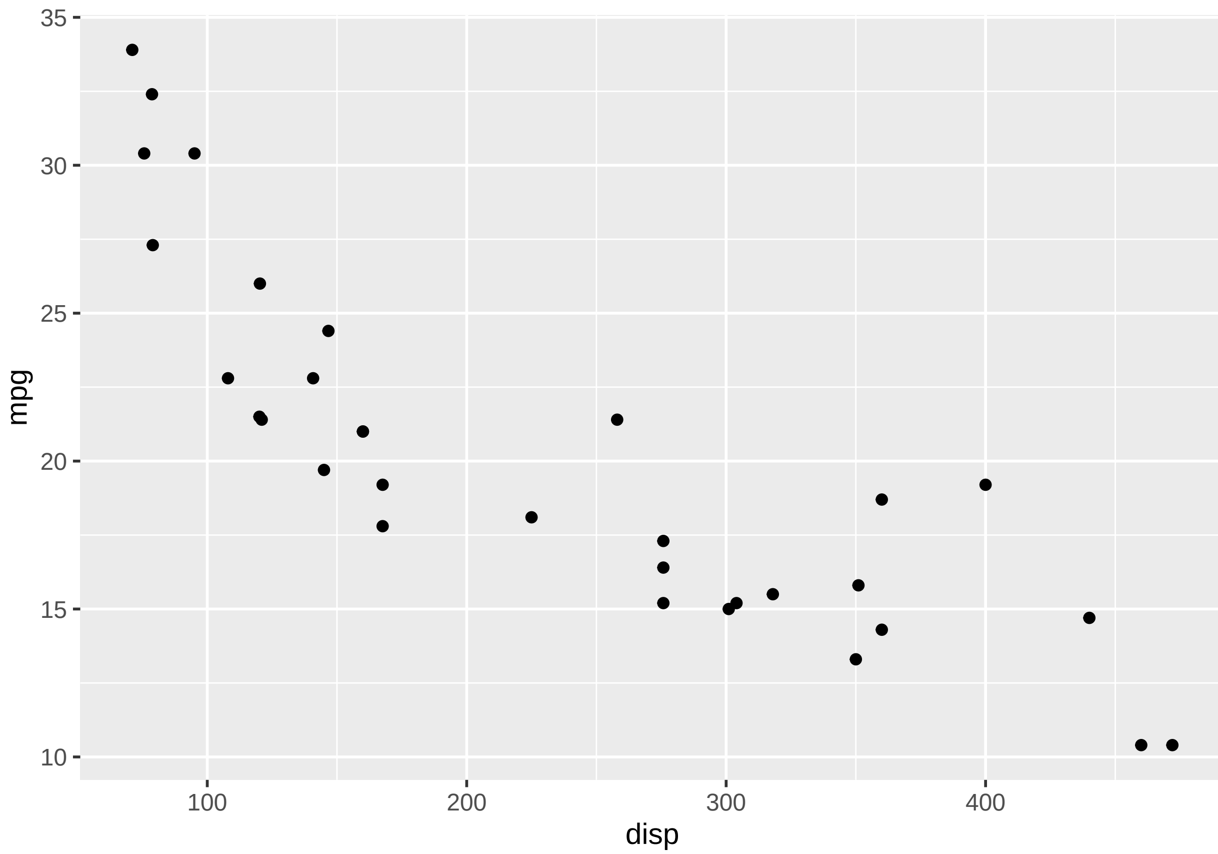

```r
# Problem 6(d)
ggplot(df, aes(x=gear, fill=factor(cyl))) + geom_bar()
```




```r
# Problem 6(e)
ggplot(df, aes(x=wt, y=mpg))+geom_point()
```

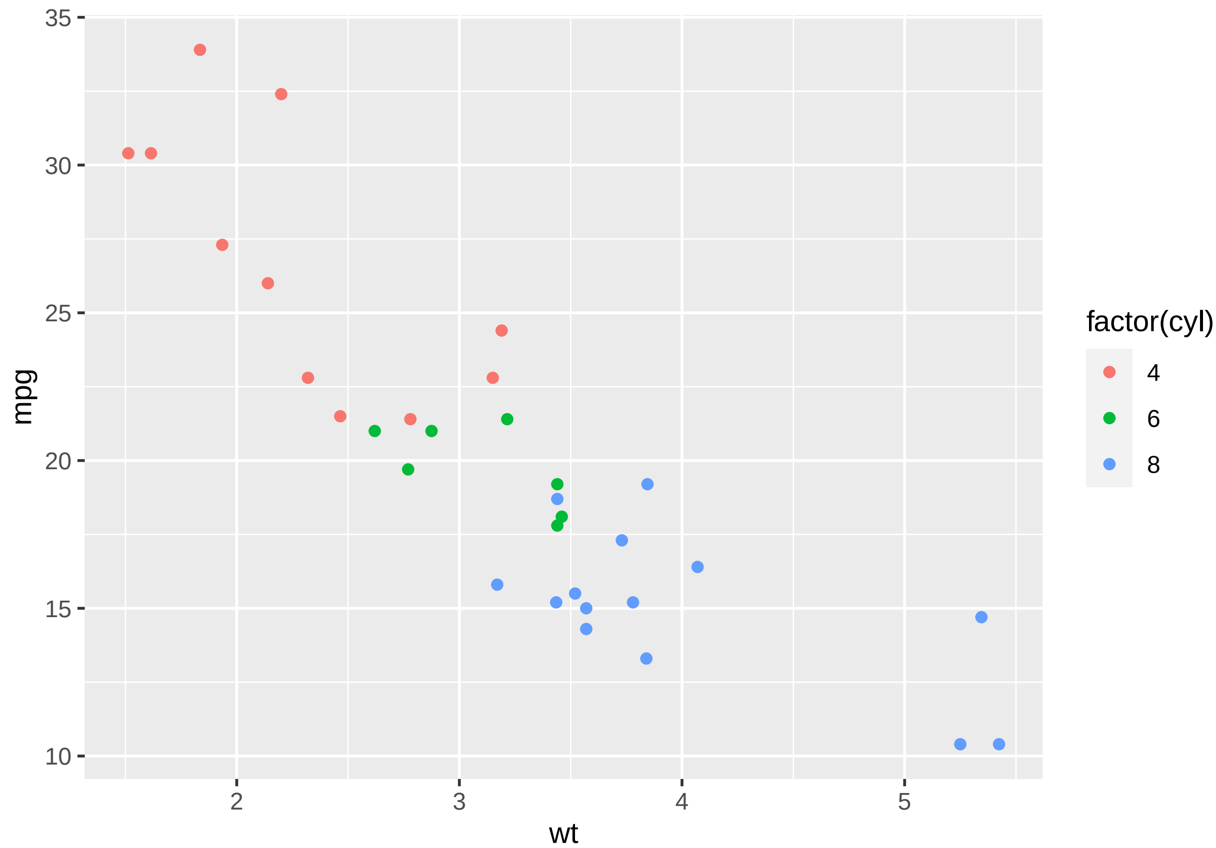



```r
# Problem 6(f)
ggplot(df, aes(x=disp, y=mpg))+geom_point()
```

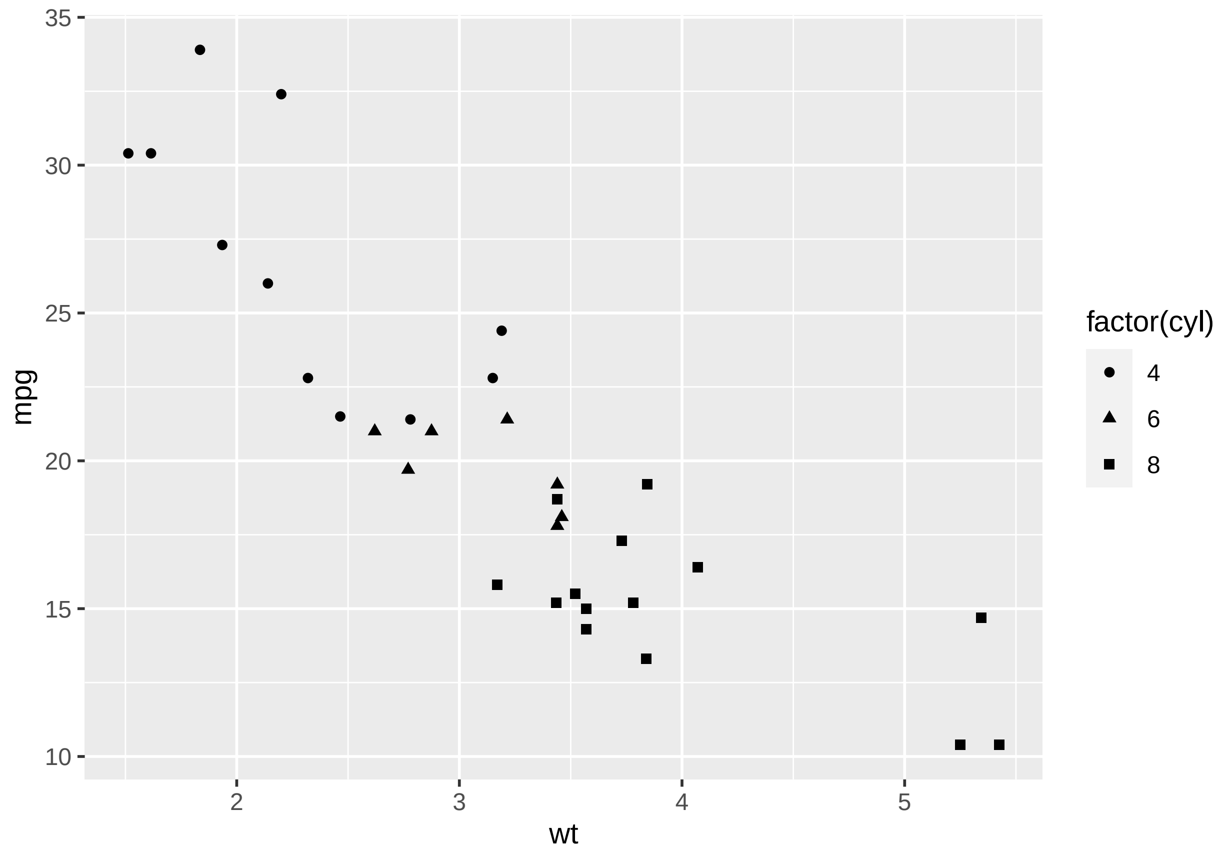



```
# There is a strong negative relationship between variables.
```

```
# Problem 6(g)
ggplot(df, aes(x=wt, y=mpg, color=factor(cyl)))+geom_point()
```




```r
# Problem 6(h)
ggplot(df, aes(x=wt, y=mpg, shape=factor(cyl)))+geom_point()
```



Problem 7

```r
## Problem 7(a)
df <- read.csv("C:/Masters - Business Analytics/Data Mining/assignment 1/gapminder.csv")

df %>%
  group_by(continent) %>%
  summarise(countries = length(unique(country)))
```

```
## # A tibble: 5 x 2
##   continent countries
##   <chr>         <int>
## 1 Africa           52
## 2 Americas         25
## 3 Asia             33
## 4 Europe           30
## 5 Oceania           2
```

```r
# Problem 7(b)
df %>%
  filter(., continent == 'Europe', year == 1997) %>%
  filter(., gdpPercap == min(.$gdpPercap))
```

```
##   country continent year lifeExp     pop gdpPercap
## 1 Albania    Europe 1997   72.95 3428038  3193.055
```

```
# Problem 7(c)
df %>%
  filter(., year >= 1980 & year <=1990) %>%
  group_by(continent) %>%
  summarise(avg_life_ex = mean(lifeExp))
```

```
## # A tibble: 5 x 2
##   continent avg_life_ex
##   <chr>           <dbl>
## 1 Africa           52.5
## 2 Americas         67.2
## 3 Asia             63.7
## 4 Europe           73.2
## 5 Oceania          74.8
```

```
# Problem 7(d)
df %>%
  group_by(country) %>%
  summarise(combined_gdp = sum(pop*gdpPercap)) %>%
  arrange(desc(combined_gdp)) %>%
  head(5)
```

```
## # A tibble: 5 x 2
##   country        combined_gdp
##   <chr>                 <dbl>
## 1 United States       7.68e13
## 2 Japan               2.54e13
## 3 China               2.04e13
## 4 Germany             1.95e13
## 5 United Kingdom      1.33e13
```

```
# Problem 7(e)
df %>%
  select(country,lifeExp,year) %>%
  filter(lifeExp >= 80)
```

```
##             country lifeExp year
## 1         Australia  80.370 2002
## 2         Australia  81.235 2007
## 3            Canada  80.653 2007
## 4            France  80.657 2007
## 5  Hong Kong, China  80.000 1997
## 6  Hong Kong, China  81.495 2002
## 7  Hong Kong, China  82.208 2007
## 8           Iceland  80.500 2002
## 9           Iceland  81.757 2007
## 10           Israel  80.745 2007
## 11            Italy  80.240 2002
## 12            Italy  80.546 2007
```

```
## 13           Japan  80.690 1997
## 14           Japan  82.000 2002
## 15           Japan  82.603 2007
## 16     New Zealand  80.204 2007
## 17          Norway  80.196 2007
## 18           Spain  80.941 2007
## 19          Sweden  80.040 2002
## 20          Sweden  80.884 2007
## 21     Switzerland  80.620 2002
## 22     Switzerland  81.701 2007
```

Problem 8

```
## Problem 8(a)
library(hflights)
data("hflights")
df <- hflights
head(df,20)
```

```
##      Year Month DayofMonth DayOfWeek DepTime ArrTime UniqueCarrier FlightNum
## 5424 2011     1          1         6    1400    1500            AA       428
## 5425 2011     1          2         7    1401    1501            AA       428
## 5426 2011     1          3         1    1352    1502            AA       428
## 5427 2011     1          4         2    1403    1513            AA       428
## 5428 2011     1          5         3    1405    1507            AA       428
## 5429 2011     1          6         4    1359    1503            AA       428
## 5430 2011     1          7         5    1359    1509            AA       428
## 5431 2011     1          8         6    1355    1454            AA       428
## 5432 2011     1          9         7    1443    1554            AA       428
## 5433 2011     1         10         1    1443    1553            AA       428
## 5434 2011     1         11         2    1429    1539            AA       428
## 5435 2011     1         12         3    1419    1515            AA       428
## 5436 2011     1         13         4    1358    1501            AA       428
## 5437 2011     1         14         5    1357    1504            AA       428
## 5438 2011     1         15         6    1359    1459            AA       428
## 5439 2011     1         16         7    1359    1509            AA       428
## 5440 2011     1         17         1    1530    1634            AA       428
## 5441 2011     1         18         2    1408    1508            AA       428
## 5442 2011     1         19         3    1356    1503            AA       428
## 5443 2011     1         20         4    1507    1622            AA       428
##      TailNum ActualElapsedTime AirTime ArrDelay DepDelay Origin Dest Distance
## 5424  N576AA                60      40      -10        0    IAH  DFW      224
## 5425  N557AA                60      45       -9        1    IAH  DFW      224
## 5426  N541AA                70      48       -8       -8    IAH  DFW      224
## 5427  N403AA                70      39        3        3    IAH  DFW      224
## 5428  N492AA                62      44       -3        5    IAH  DFW      224
## 5429  N262AA                64      45       -7       -1    IAH  DFW      224
## 5430  N493AA                70      43       -1       -1    IAH  DFW      224
## 5431  N477AA                59      40      -16       -5    IAH  DFW      224
## 5432  N476AA                71      41       44       43    IAH  DFW      224
## 5433  N504AA                70      45       43       43    IAH  DFW      224
## 5434  N565AA                70      42       29       29    IAH  DFW      224
```

```
## 5435  N577AA                56      41       5      19    IAH  DFW      224
## 5436  N476AA                63      44      -9      -2    IAH  DFW      224
## 5437  N552AA                67      47      -6      -3    IAH  DFW      224
## 5438  N462AA                60      44     -11      -1    IAH  DFW      224
## 5439  N555AA                70      41      -1      -1    IAH  DFW      224
## 5440  N518AA                64      48      84      90    IAH  DFW      224
## 5441  N507AA                60      42      -2       8    IAH  DFW      224
## 5442  N523AA                67      46      -7      -4    IAH  DFW      224
## 5443  N425AA                75      42      72      67    IAH  DFW      224
##      TaxiIn TaxiOut Cancelled CancellationCode Diverted
## 5424      7      13         0                         0
## 5425      6       9         0                         0
## 5426      5      17         0                         0
## 5427      9      22         0                         0
## 5428      9       9         0                         0
## 5429      6      13         0                         0
## 5430     12      15         0                         0
## 5431      7      12         0                         0
## 5432      8      22         0                         0
## 5433      6      19         0                         0
## 5434      8      20         0                         0
## 5435      4      11         0                         0
## 5436      6      13         0                         0
## 5437      5      15         0                         0
## 5438      6      10         0                         0
## 5439     12      17         0                         0
## 5440      8       8         0                         0
## 5441      7      11         0                         0
## 5442     10      11         0                         0
## 5443      9      24         0                         0
```

```r
# Problem 8(b)
df %>%
  filter(., DayofMonth == 1, Month == 1) %>%
  head() #displaying only limited rows to reduce number of pages
```

```
##   Year Month DayofMonth DayOfWeek DepTime ArrTime UniqueCarrier FlightNum
## 1 2011     1          1         6    1400    1500            AA       428
## 2 2011     1          1         6     728     840            AA       460
## 3 2011     1          1         6    1631    1736            AA      1121
## 4 2011     1          1         6    1756    2112            AA      1294
## 5 2011     1          1         6    1012    1347            AA      1700
## 6 2011     1          1         6    1211    1325            AA      1820
##   TailNum ActualElapsedTime AirTime ArrDelay DepDelay Origin Dest Distance
## 1  N576AA                60      40      -10        0    IAH  DFW      224
## 2  N520AA                72      41        5        8    IAH  DFW      224
## 3  N4WVAA                65      37       -9        1    IAH  DFW      224
## 4  N3DGAA               136     113       -3        1    IAH  MIA      964
## 5  N3DAAA               155     117        7       -8    IAH  MIA      964
## 6  N593AA                74      39       15        6    IAH  DFW      224
##   TaxiIn TaxiOut Cancelled CancellationCode Diverted
## 1      7      13         0                         0
## 2      6      25         0                         0
## 3     16      12         0                         0
```

```
## 4      9      14          0                          0
## 5     12      26          0                          0
## 6      6      29          0                          0
```

```r
# Problem 8(c)
df %>%
  filter(., UniqueCarrier=='AA' | UniqueCarrier=='UA') %>%
  head()
```

```
##   Year Month DayofMonth DayOfWeek DepTime ArrTime UniqueCarrier FlightNum
## 1 2011     1          1         6    1400    1500            AA       428
## 2 2011     1          2         7    1401    1501            AA       428
## 3 2011     1          3         1    1352    1502            AA       428
## 4 2011     1          4         2    1403    1513            AA       428
## 5 2011     1          5         3    1405    1507            AA       428
## 6 2011     1          6         4    1359    1503            AA       428
##   TailNum ActualElapsedTime AirTime ArrDelay DepDelay Origin Dest Distance
## 1  N576AA                60      40      -10        0    IAH  DFW      224
## 2  N557AA                60      45       -9        1    IAH  DFW      224
## 3  N541AA                70      48       -8       -8    IAH  DFW      224
## 4  N403AA                70      39        3        3    IAH  DFW      224
## 5  N492AA                62      44       -3        5    IAH  DFW      224
## 6  N262AA                64      45       -7       -1    IAH  DFW      224
##   TaxiIn TaxiOut Cancelled CancellationCode Diverted
## 1      7      13         0                         0
## 2      6       9         0                         0
## 3      5      17         0                         0
## 4      9      22         0                         0
## 5      9       9         0                         0
## 6      6      13         0                         0
```

```r
# Problem 8(d)
df %>%
  select(Year, Month, DayofMonth, contains("Taxi"), contains("Delay")) %>%
  head()
```

```
##      Year Month DayofMonth TaxiIn TaxiOut ArrDelay DepDelay
## 5424 2011     1          1      7      13      -10        0
## 5425 2011     1          2      6       9       -9        1
## 5426 2011     1          3      5      17       -8       -8
## 5427 2011     1          4      9      22        3        3
## 5428 2011     1          5      9       9       -3        5
## 5429 2011     1          6      6      13       -7       -1
```

```r
# Problem 8(e)
df %>%
  select(departure_time =DepTime, arrival_time=ArrTime, flight_number=FlightNum) %>%
  head()
```

```
##      departure_time arrival_time flight_number
## 5424           1400         1500           428
## 5425           1401         1501           428
```

```
## 5426           1352         1502           428
## 5427           1403         1513           428
## 5428           1405         1507           428
## 5429           1359         1503           428
```

```
# Problem 8(f)
df %>%
  filter(., DepDelay > 60) %>%
  select(UniqueCarrier) %>%
  unique()
```

```
##     UniqueCarrier
## 1              AA
## 10             AS
## 11             B6
## 16             CO
## 179            DL
## 185            OO
## 213            UA
## 215            US
## 221            WN
## 244            EV
## 255            F9
## 256            FL
## 258            MQ
## 419            XE
```

```
# Problem 8(g)
df %>%
  select(UniqueCarrier, DepDelay) %>%
  arrange(desc(DepDelay)) %>%
  head()
```

```
##   UniqueCarrier DepDelay
## 1            CO      981
## 2            AA      970
## 3            MQ      931
## 4            UA      869
## 5            MQ      814
## 6            MQ      803
```

Problem 9. Consider the following data set:

record number	income	student	credit-rating	buys-computer
1	high	no	fair	no
2	high	no	excellent	no
3	low	no	excellent	yes
4	medium	no	fair	no
5	low	yes	fair	no
6	low	yes	excellent	yes
7	low	no	excellent	yes
8	medium	yes	fair	yes
9	low	yes	fair	no
10	medium	yes	fair	yes
11	medium	yes	excellent	yes
12	medium	no	excellent	no
13	high	yes	fair	no
14	medium	yes	excellent	yes

(a) Using the 1-rule method discussed in class, find the relevant sets of classification rules for the target buys-computer by testing each of the input attributes income, student, and credit-rating. Which of these three sets of rules has the lowest misclassification rate?
(b) Considering "buy-computer" as the target variable, which of the attributes would you select as the root in a decision tree that is constructed using the Gini index impurity measure?
(c) Use the Gini index impurity measure and construct the full decision tree for this data set.
(d) Using your decision tree, provide two strong decision rules that we can use to predict whether a student is going to buy computer or not. Justify your choice.
(e) What is the accuracy of your decision tree model on the training examples

__**Solutions:**__

(a) Let's start with "**Income**"
If income = high, then buys_computer = no (3 out of 3)
If income = medium, then buys_computer = yes (4 out of 6)
If income = low, then buys_computer = yes (3 out of 5)
Proceeding this way, we get total misclassification as **4/14**

Taking, **"Student"** variable,
If student = no, then buys_computer = no (4 of 6)
If student = yes, then buys_computer = yes (5 of 8)
Proceeding this way, the total misclassifications are **5/14**

Considering **credit-rating**,
If credit-rating = fair, then buys_computer = no (5 of 7)
If credit-rating = excellent, then buys_computer = yes (5 of 7)
Proceeding this way, we get a total misclassifications of **4/14**

If we are strictly sticking with just these 3 categories of classifications and not considering the further sub classifications, then going with **“credit-rating”** makes most sense since it has the minimum misclassifications tied along with “Income” and has lesser categories than Income. Hence the model building will be simpler and computation time will be lesser.

(b) Using “Income”:

Gini(I) = $(3/14) \times (1-(0)^2-(1)^2) + (6/14) \times (1-(4/6)^2-(2/6)^2) + (5/14) \times (1-(3/5)^2-(2/5)^2) =$

$= (6/14)*(0.444) + (5/14)*(0.48)$

$\mathbf{= 0.362}$

Using “Student”:

Gini(s) = $(6/14)*(1-(2/6)^2-(4/6)^2) + (8/14)*(1-(5/8)^2 -(3/8)^2)$

$= (6/14)*(0.444) + (8/14)*(0.469)$

$= 0.458$

Using “credit-rating”:

Gini(c) = $(7/14)*(1-(2/7)^2-(5/7)^2) + (1/2)*(1 - (5/7)^2 - (2/7)^2)$

$= (1/2)*(0.408) + (1/2)*(0.408)$

$= 0.408$

Going by Gini values, **“Income”** seems to be the best attribute for the roots of decision tree.

(c) We’ll take “Income’ as the base root, for the next nodes:

For **Income = low**, if **credit_rating = fair**, then **buys_computer = no,**
If **credit_rating = excellent**, then **buys_computer = yes**

For **Income = medium**, if **student = yes**, then **buys_computer = yes**,
If **student = no**, the **buys_computer = no**.

We haven’t considered the “student” attribute for income = low because we won’t be getting pure nodes, hence the Gini will without doubt be higher than if we split using “credit”. Similarly for the other node we’ve taken “student” attribute to split.

After splitting We can observe that are left with completely pure nodes, hence the **Gini impurity after splitting to max depth is 0**, which is the lowest and best possible value for any dataset.

(d) There are 3 rules which can give us an accurate prediction, them being:

- If income = high, buys_computer = No.

- **If Income = low and credit_rating=fair, then, buys_computer = No.**
  If income=low and credit_rating = excellent, then, buys_computer = Yes.

- **If Income = medium and student = yes, then, buys_computer = Yes.**
  If Income = medium and student = no, then, buys_computer = No.

  While all the above 3 rules hold true with respect to the given dataset, the last 2 seems to be the strongest of the 3 since they take multiple attributes into consideration. But to be precise we need to check the significance/importance of each attribute and then make a decision based on the results.

(e) Our decision tree can accurately classify all of the given records, hence if we use the training data to make predictions, the **accuracy of the DT will be 100%**