Statistical Data Mining II - Homework 1

Ezgi Karaesmen

February 24, 2016

Exercise 1: Preprocessing the bodyfat dataset

`bodyfat` dataset consists of 15 variables, which are different body measurements taken from 252 subjects.

```
library(DAAG)
library(lattice)
library(MASS)
library(ggplot2)
library(gridExtra)
head(bodyfat, 3)
```

```
##   density bodyfat age weight height neck chest abdomen  hip thigh knee
## 1  1.0708    12.3  23 154.25  67.75 36.2  93.1    85.2 94.5  59.0 37.3
## 2  1.0853     6.1  22 173.25  72.25 38.5  93.6    83.0 98.7  58.7 37.3
## 3  1.0414    25.3  22 154.00  66.25 34.0  95.8    87.9 99.2  59.6 38.9
##   ankle biceps forearm wrist
## 1  21.9   32.0    27.4  17.1
## 2  23.4   30.5    28.9  18.2
## 3  24.0   28.8    25.2  16.6
```

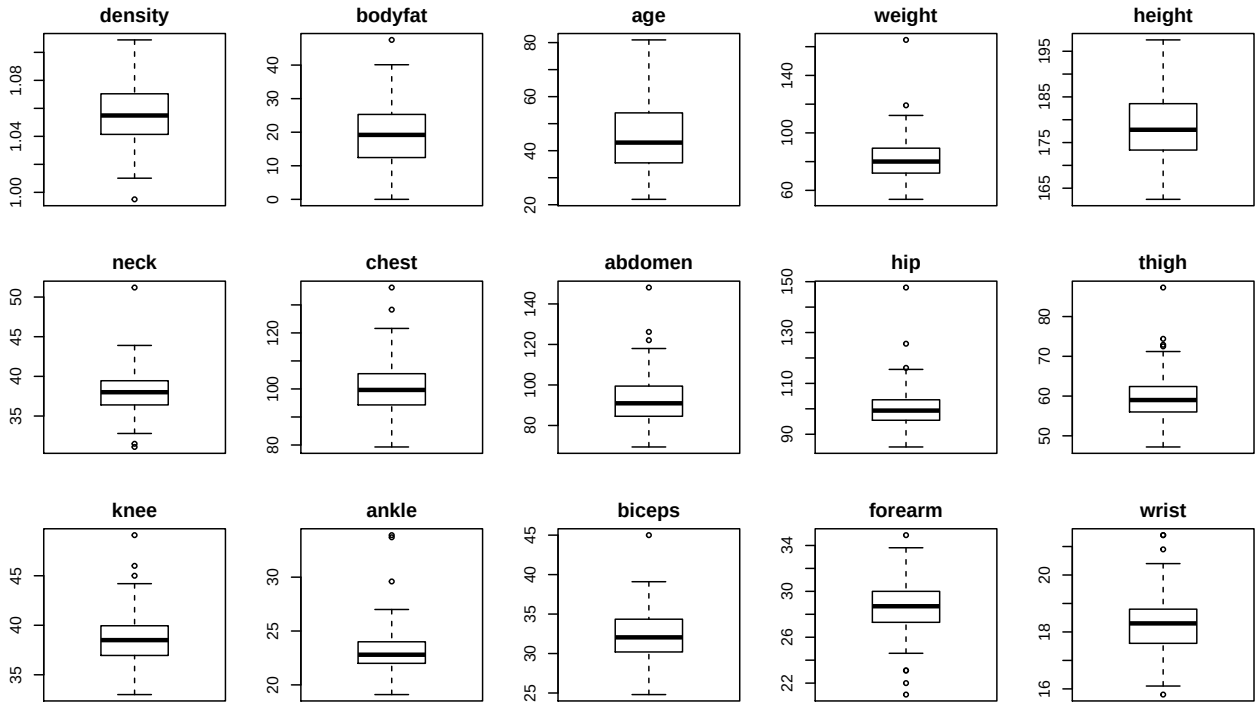
As indicated in the `body_fat_details.pdf` file, most of the measurements in the dataset were reported in metric system, including the `density` variable with gr/cm^3 scale. However, variables `height` and `weight` were entered in imperial system. For a more accurate analysis, `height`, and `weight` variables were converted into cm and kg scales respectively.

```
## Turn height from inch into cm
# 1 inch = 2.54 cm
bodyfat$height <- bodyfat$height*2.54

## Turn weight from lbs into grams
# 1 pound = 0.453592 kg
bodyfat$weight <- bodyfat$weight*0.453592

head(bodyfat[, c("weight", "height")])
```

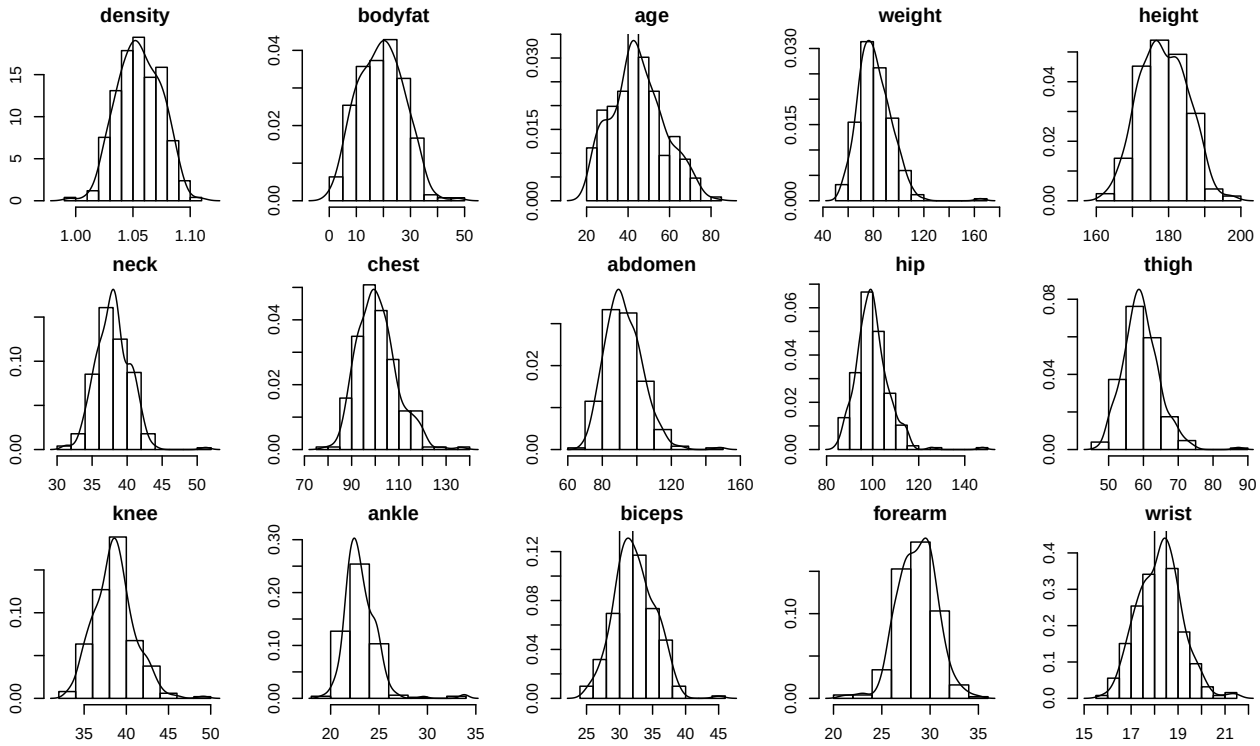
```
##     weight  height
## 1 69.96657 172.085
## 2 78.58481 183.515
## 3 69.85317 168.275
## 4 83.80112 183.515
## 5 83.57433 180.975
## 6 95.36772 189.865
```

In order to explore the distribution profile of the variables and detect possible outliers, every variable of the data was visualized with box plots and histograms.

```
par(mfrow = c(3,5), mar=(c(2,2,2,2)))
## boxplots
for(i in 1:ncol(bodyfat)){boxplot(bodyfat[,i], main=colnames(bodyfat)[i])}
```

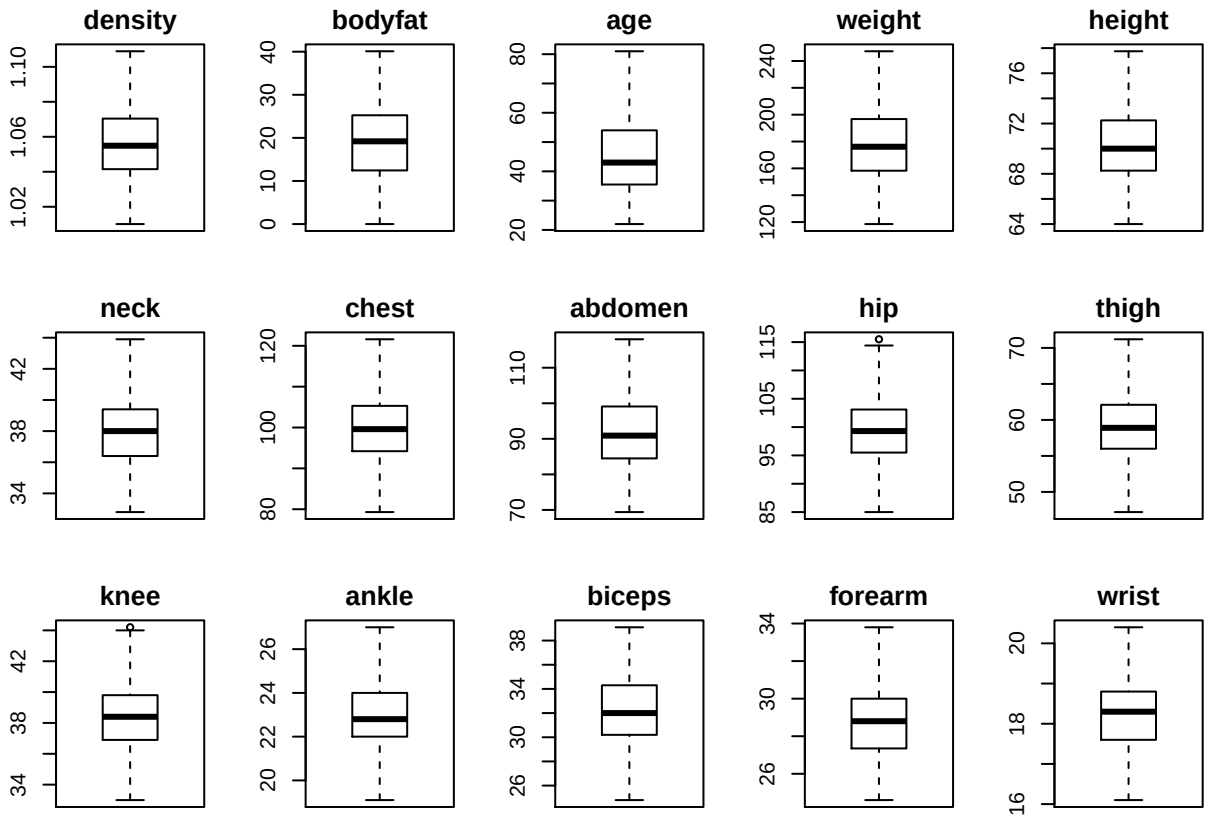



```
den=NULL
for(i in 1:ncol(bodyfat)){
  den <- density(bodyfat[,i])
  hist(bodyfat[,i], probability = T, xlim= range(den$x), ylim=range(den$y),
       main=colnames(bodyfat)[i])
  lines(den)}
```

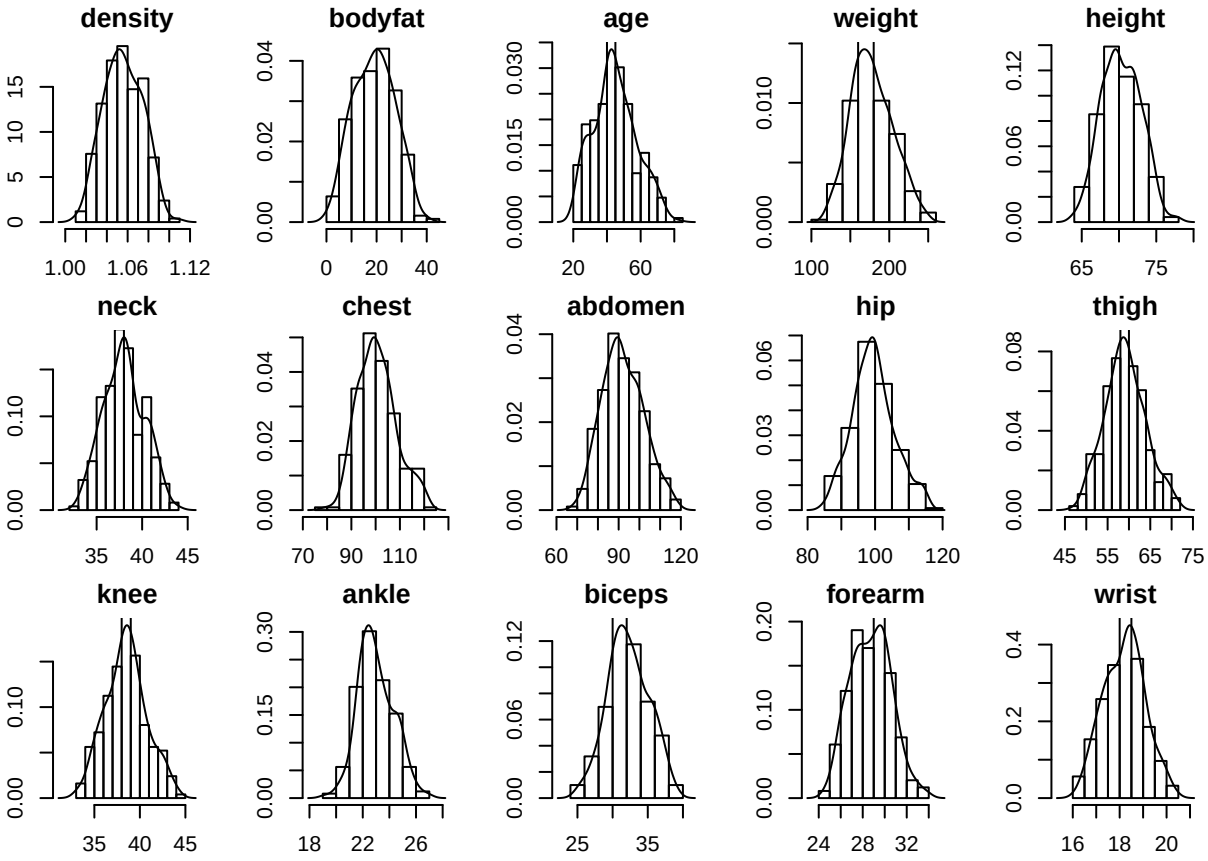


Majority of the variable distributions have a distribution close to normal, whereas some variables such as `weight, hip, thigh, knee and ankle` show a longer upper tail. Furthermore, outliers were detected for majority of the variables, and the left-shift seen in above mentioned variable distributions is likely due to the presence of outliers. Hence, outliers were set as `NA`s and the variables were re-visualized.

```
# set outliers to NA's
for(i in 1:ncol(bodyfat)){
  outliers <- which(bodyfat[,i] %in% boxplot.stats(bodyfat[,i])$out == TRUE)
  bodyfat_o2na[outliers,i] <- NA
}

# boxplot and histogram with densities
par(mfrow = c(3,5), mar=(c(2,2,2,2)))
for(i in 1:ncol(bodyfat_o2na)){boxplot(bodyfat_o2na[,i],main=colnames(bodyfat)[i])}
```



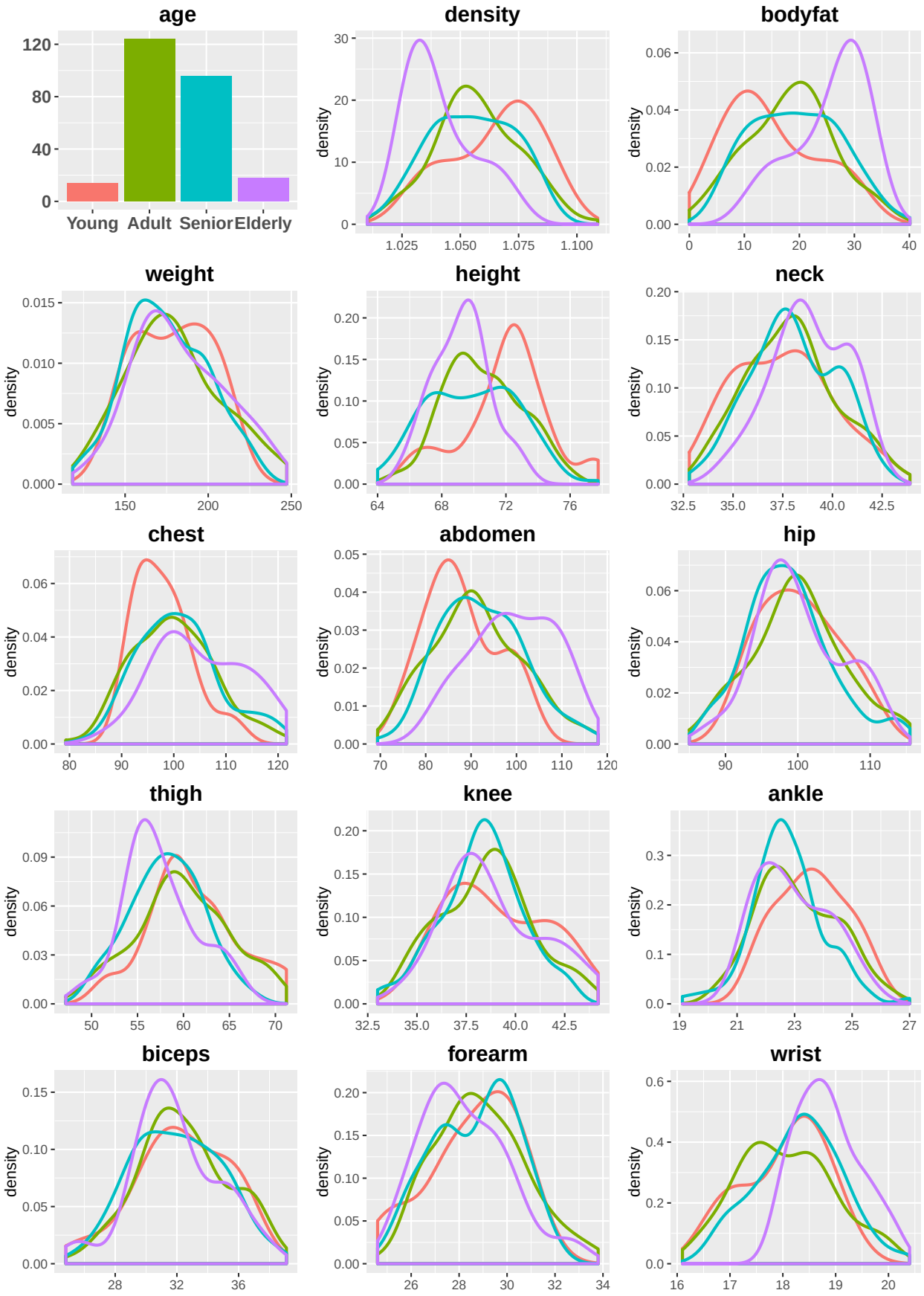

```
den=NULL
for(i in 1:ncol(bodyfat)){
  den <- density(bodyfat_o2na[,i], na.rm=T)
  hist(bodyfat_o2na[,i], probability = T, xlim= range(den$x), ylim=range(den$y),
       main=colnames(bodyfat)[i])
  lines(den)}
```



Upon outlier removal, no outliers were seen on the box plots, indicating the removal was successful. Furthermore, skewed distributions such as `weight, hip, thigh, knee` and `ankle` show a much more symmetrical distribution upon outlier removal.

In order to explore different groups with in the data, subjects were split into four groups according to their age. Densities of different variables were then visualized using `ggplot`.

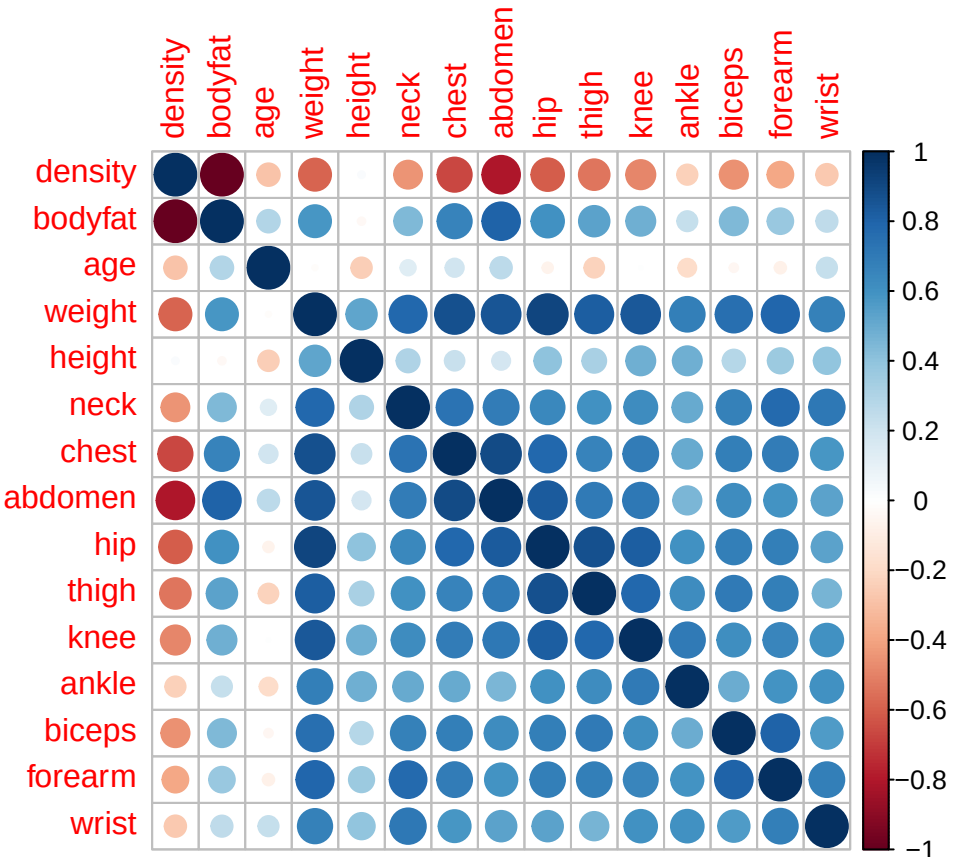
```
bdf <- bodyfat_o2na
bdf[["age"]] <- ordered(cut(bodyfat_o2na[["age"]], c(15 , 25, 45 , 65, 100)),
                                labels=c("Young", "Adult", "Senior", "Elderly"))
```


age
120
80
40
0
Young
Adult
Senior
Elderly
density
density
30
20
10
0
1.025
1.050
1.075
1.100
bodyfat
density
0.06
0.04
0.02
0.00
0
10
20
30
40
weight
density
0.015
0.010
0.005
0.000
150
200
250
height
density
0.20
0.15
0.10
0.05
0.00
64
68
72
76
neck
density
0.20
0.15
0.10
0.05
0.00
32.5
35.0
37.5
40.0
42.5
chest
density
0.06
0.04
0.02
0.00
80
90
100
110
120
abdomen
density
0.05
0.04
0.03
0.02
0.01
0.00
70
80
90
100
110
120
hip
density
0.06
0.04
0.02
0.00
90
100
110
thigh
density
0.09
0.06
0.03
0.00
50
55
60
65
70
knee
density
0.20
0.15
0.10
0.05
0.00
32.5
35.0
37.5
40.0
42.5
ankle
density
0.3
0.2
0.1
0.0
19
21
23
25
27
biceps
density
0.15
0.10
0.05
0.00
28
32
36
forearm
density
0.20
0.15
0.10
0.05
0.00
26
28
30
32
34
wrist
density
0.6
0.4
0.2
0.0
16
17
18
19
20

`Density` and `bodyfat` variables show different density profiles for `young` and `elderly`, where as `adult` and `senior` subjects do not. Additionally `density` and `bodyfat` show opposite densities, indicating the inverse correlation between there variables. This is expected as `density` is a denominator of the `bodyfat`. Overall, age groups do not seem to be distributed differently for other variables. It is important to note that grouping the age variable for different intervals will effect the outcome.

Another approach to explore the relationship between variables and detect abnormalities is to produce a correlation matrix.

```
library(corrplot)
cor_bf <- cor(as.matrix(bodyfat_o2na), use="complete.obs", method = "pearson")
corrplot(cor_bf, method = "circle")
```



The correlation matrix visualizes the correlation between variables, where -1 (dark red) and 1 (dark blue) indicate strong negative and positive correlation respectively. As concluded from the density plots shown previously, density and body fat show strong inverse correlation. Furthermore, age only shows some correlation with density, bodyfat and several other variables. Overall majority of the variables show a positive correlation, which is expected as increase in body weight would result in the increase of body volume, hence the circumference measurements of different body parts.

Bodyfat data is a relatively clean dataset and upon height/weight re-scaling and outlier removal data would be ready for analysis.

```
save(bodyfat_o2na, file="./clean_data.RData")
```

Exercise 2: Computing Jaccard and Cosine distances

Jaccard similarity coefficient is a statistic used for quantifying the similarity between samples and is defined as:

$$J = \frac{M_{1,1}}{(M_{1,0} + M_{0,1} - M_{1,1})}$$

where,

- $M_{1,1}$ = total number of attributes positive for both objects
- $M_{1,0}$ = total number of attributes 1 for i and 0 for j
- $M_{0,1}$ = total number of attributes 1 for i and 0 for j

In other terms the numerator and the denominator of the Jaccard similarity is the number of elements in the intersection and the union of two object vectors respectively. Jaccard distance on the other hand, measures dissimilarity between samples, and is complementary to the Jaccard coefficient and is obtained by subtracting the Jaccard coefficient from 1:

$$d_J = 1 - J$$

Cosine similarity a distance metric that measures the cosine of the angle between the object vectors, and therefore can judge the orientation of the vector. Cosine similarity is defined by:

$$S_C(A, B) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}$$

where A and B are object vectors. Similar to the Jaccard distance, cosine distance can be obtained by subtracting the cosine similarity from 1:

$$D_C(A, B) = 1 - S_C(A, B)$$

However, this is not a proper distance metric as it does not have the triangle inequality property and it violates the coincidence axiom [*Wikipedia*]. Therefore, for this exercise the cosine similarity will be computed and interpreted in terms of distance/dissimilarity. With the above given information, one can write the `jaccard` and `cosine` functions:

```
# Jaccard distance
jaccard <- function(x1, x2){
  aib <- sum(x1[x1==x2])
  aub <- sum(x1+x2)-aib
  round(1-(aib/aub), 3)
}

# cosine similarity
cosine <- function(x1, x2){
  ab <- x1*x2
  ab <- sum(ab[!is.na(ab)])
  AA <- x1^2
```

```
  AA <- sum(AA[!is.na(AA)])
  BB <- x2^2
  BB <- sum(BB[!is.na(BB)])
  round(ab/(sqrt(AA)*sqrt(BB)), 3)
}
```

(a) Treat the utility matrix as Boolean and compute the Jaccard distance, and the cosine distance between users.

The original and Boolean matrices were produced with `R` and named `mat0` and `mat1` respectively. Then the Jaccard distance and cosine similarity between three users were calculated for the Boolean matrix.

```
# original matrix (mat0)
A=c(4, 5, NA, 5, 1, NA, 3, 2)
B=c(NA, 3, 4, 3, 1, 2, 1, NA)
C=c(2, NA, 1, 3, NA, 4, 5, 3)

mat0 <- as.matrix(rbind(A,B,C), dimnames = list(c("A","B", "C")))
colnames(mat0) <- c(letters[1:8])
mat0
```

```
##    a  b  c d  e  f g  h
## A  4  5 NA 5  1 NA 3  2
## B NA  3  4 3  1  2 1 NA
## C  2 NA  1 3 NA  4 5  3
```

```
#  boolean matrix 1 (mat1)
A1=c(T, T, F, T, T, F, T, T)
B1=c(F, T, T, T, T, T, T, F)
C1=c(T, F, T, T, F, T, T, T)

mat1 <- as.matrix(rbind(A1,B1,C1), dimnames = list(c("A","B", "C")))
colnames(mat1) <- c(letters[1:8])
mat1
```

```
##        a     b     c    d     e     f    g     h
## A1  TRUE  TRUE FALSE TRUE  TRUE FALSE TRUE  TRUE
## B1 FALSE  TRUE  TRUE TRUE  TRUE  TRUE TRUE FALSE
## C1  TRUE FALSE  TRUE TRUE FALSE  TRUE TRUE  TRUE
```

```
#Jaccard distances for mat1
ab.j1 <- jaccard(A1, B1)
ac.j1 <- jaccard(A1, C1)
bc.j1 <- jaccard(B1, C1)
list("Jaccard distances between users for boolean matrix mat1",
      matrix(c(ab.j1, 0, ac.j1, bc.j1), nrow = 2, ncol = 2,
             dimnames=list(c("A","B"),c("B", "C") )))
```

```
## [[1]]
## [1] "Jaccard distances between users for boolean matrix mat1"
##
```

```
## [[2]]
##     B   C
## A 0.5 0.5
## B 0.0 0.5
```

```
#Cosine similarities for mat1
ab.c1 <- cosine(A1, B1)
ac.c1 <- cosine(A1, C1)
bc.c1 <- cosine(B1, C1)
list("Cosine similarities between users for boolean matrix mat1",
      matrix(c(ab.c1, 1, ac.c1, bc.c1), nrow = 2, ncol = 2,
             dimnames=list(c("A","B"),c("B", "C") )))
```

```
## [[1]]
## [1] "Cosine similarities between users for boolean matrix mat1"
##
## [[2]]
##       B     C
## A 0.667 0.667
## B 1.000 0.667
```

It can be seen that the distances between users for mat1 are calculated as equal for both of the distance measures. This is expected, since in this binary format all users have same number of intersecting elements. It is important to note that this method ignores the actual ratings as it only takes into account the common items that were rated by the users and not the rating given.

(b) Try a different discretization: treat ratings 3,4,5 as 1, and ratings 1, 2, and blank as 0. Compute the Jaccard distance and cosine distance and compare to that of part A.

The utility matrix produced in part (a) with different discretization was produced with R and named mat2. Jaccard and cosine distances between mat2 users were computed.

```
#   boolean matrix 2 (mat2)
A2=c(1, 1, 0, 1, 0, 0, 1, 0)
B2=c(0, 1, 1, 1, 0, 0, 0, 0)
C2=c(0, 0, 0, 1, 0, 1, 1, 1)

mat2 <- as.matrix(rbind(A2,B2,C2), dimnames = list(c("A","B", "C")))
colnames(mat1) <- c(letters[1:8])
mat2
```

```
##    [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
## A2    1    1    0    1    0    0    1    0
## B2    0    1    1    1    0    0    0    0
## C2    0    0    0    1    0    1    1    1
```

```
#Jaccard distances for mat2
ab.j2 <- jaccard(A2, B2)
ac.j2 <- jaccard(A2, C2)
bc.j2 <- jaccard(B2, C2)
```

```
list("Jaccard distances between users for boolean matrix mat2",
      matrix(c(ab.j2, 0, ac.j2, bc.j2), nrow = 2, ncol = 2,
              dimnames=list(c("A","B"),c("B", "C") )))
```

```
## [[1]]
## [1] "Jaccard distances between users for boolean matrix mat2"
##
## [[2]]
##     B     C
## A 0.6 0.667
## B 0.0 0.833
```

```
#Cosine similarities for mat1
ab.c2 <- cosine(A2, B2)
ac.c2 <- cosine(A2, C2)
bc.c2 <- cosine(B2, C2)
list("Cosine similarities between users for boolean matrix mat2",
      matrix(c(ab.c2, 1, ac.c2, bc.c2), nrow = 2, ncol = 2,
              dimnames=list(c("A","B"),c("B", "C") )))
```

```
## [[1]]
## [1] "Cosine similarities between users for boolean matrix mat2"
##
## [[2]]
##       B     C
## A 0.577 0.500
## B 1.000 0.289
```

In this exercise distances were computed by taking the "high ratings"" into account. As any rating above 2 was taken as 1 and a that are 2, 1 or blank was determined as 0, meaning that users with similar high ratings will be less distant and vice versa. Therefore distance statistics are different than exercise (b). Jaccard distance is the smallest and cosine similarity is the highest between users A and B, meaning that these users have the most similar rating profile between all three. On the other hand, Jaccard distance is the greatest and cosine similarity is the smallest between users B and C, meaning that these users have the most different rating profile between all three. Overall users neither of three pairs is very close.

(c) Normalize the matrix by subtracting from each nonblank entry the average value for its user. Using this matrix, compute the cosine distance between each pair of users.

Original matrix with the actual ratings was normalized in `R`. Note that matrix contains `NA` and not `0` to indicate the blank space.

```
#Normalize mat0
norm <- function(x){
  m <- mean(x, na.rm=T)
  x - m
}

# normalized mat 0
mat0_n <- t(round(apply(mat0, 1, norm),3))
mat0_n
```

```
##        a     b      c     d      e      f      g      h
## A  0.667 1.667     NA 1.667 -2.333     NA -0.333 -1.333
## B     NA 0.667  1.667 0.667 -1.333 -0.333 -1.333     NA
## C -1.000    NA -2.000 0.000     NA  1.000  2.000  0.000
```

```
An <- mat0_n[1,]
Bn <- mat0_n[2,]
Cn <- mat0_n[3,]

#Cosine similarities for mat0_n
ab.cn <- cosine(An, Bn)
ac.cn <- cosine(An, Cn)
bc.cn <- cosine(Bn, Cn)
list("Cosine similarities between users for normalized matrix mat0_n",
      matrix(c(ab.cn, 1, ac.cn, bc.cn), nrow = 2, ncol = 2,
              dimnames=list(c("A","B"),c("B", "C") )))
```

```
## [[1]]
## [1] "Cosine similarities between users for normalized matrix mat0_n"
##
## [[2]]
##       B      C
## A 0.584 -0.115
## B 1.000 -0.740
```

Normalization turns low ratings into negative numbers and high ratings into positive numbers. Hence, users with opposite ratings of the objects they rated in common will have completely different vector orientations, setting the vectors as far apart as possible. However, users with similar ratings for the objects they rated in common will have a relatively small angle between them. It can be seen that users A and B have the most similar profile, where B and C have the most different and A & C are dissimilar as well.

Exercise 3 Association Rules with `arules` package

(a) Visualize the data using histograms of the different variables in the data set. Transform the data into a binary incidence matrix, and justify the choices you make in grouping categories.

Boston data contains 14 different measurements and statistics from 500 towns in Boston area. In order to explore the distribution of the variables across towns, each variable was visualized with box plots and histograms.

```
library(ElemStatLearn)
```

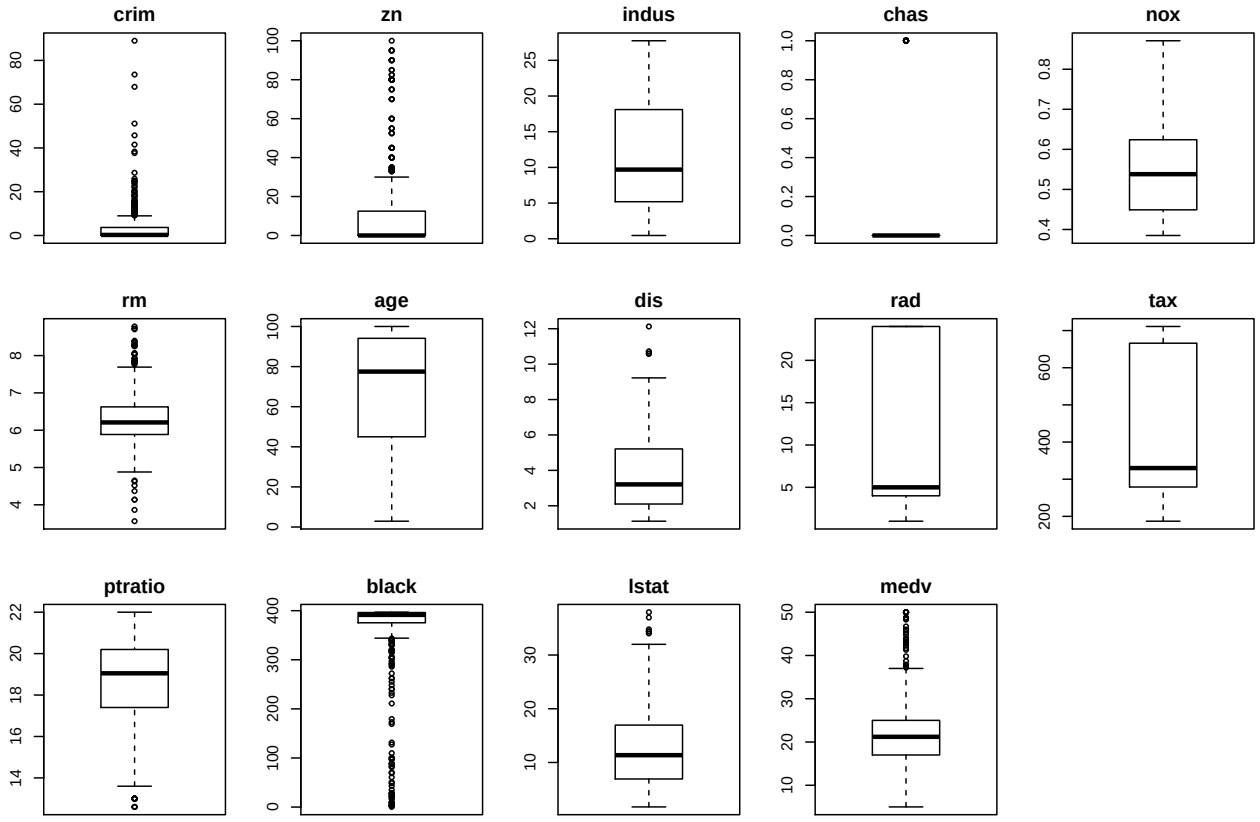
```
##
## Attaching package: 'ElemStatLearn'
```

```
## The following object is masked from 'package:DAAG':
##
##     ozone
```

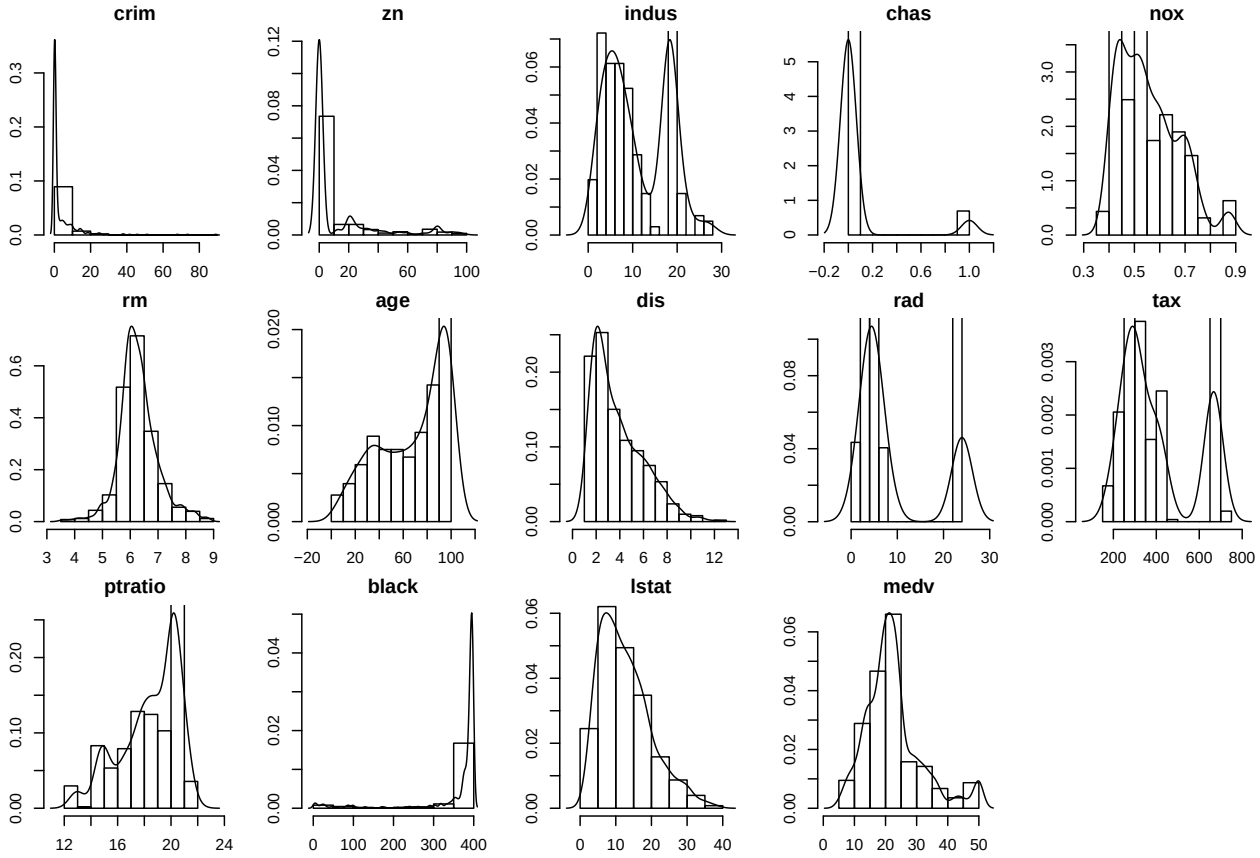
```
library(MASS)
data(Boston)
head(Boston)
```

```
##      crim zn indus chas   nox    rm  age    dis rad tax ptratio  black
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12
##   lstat medv
## 1  4.98 24.0
## 2  9.14 21.6
## 3  4.03 34.7
## 4  2.94 33.4
## 5  5.33 36.2
## 6  5.21 28.7
```

```
## boxplots
par(mfrow = c(3,5), mar=(c(2,2,2,2)))
for(i in 1:ncol(Boston)){boxplot(Boston[,i], main=colnames(Boston)[i])}

## histograms
par(mfrow = c(3,5), mar=(c(2,2,2,2)))
```

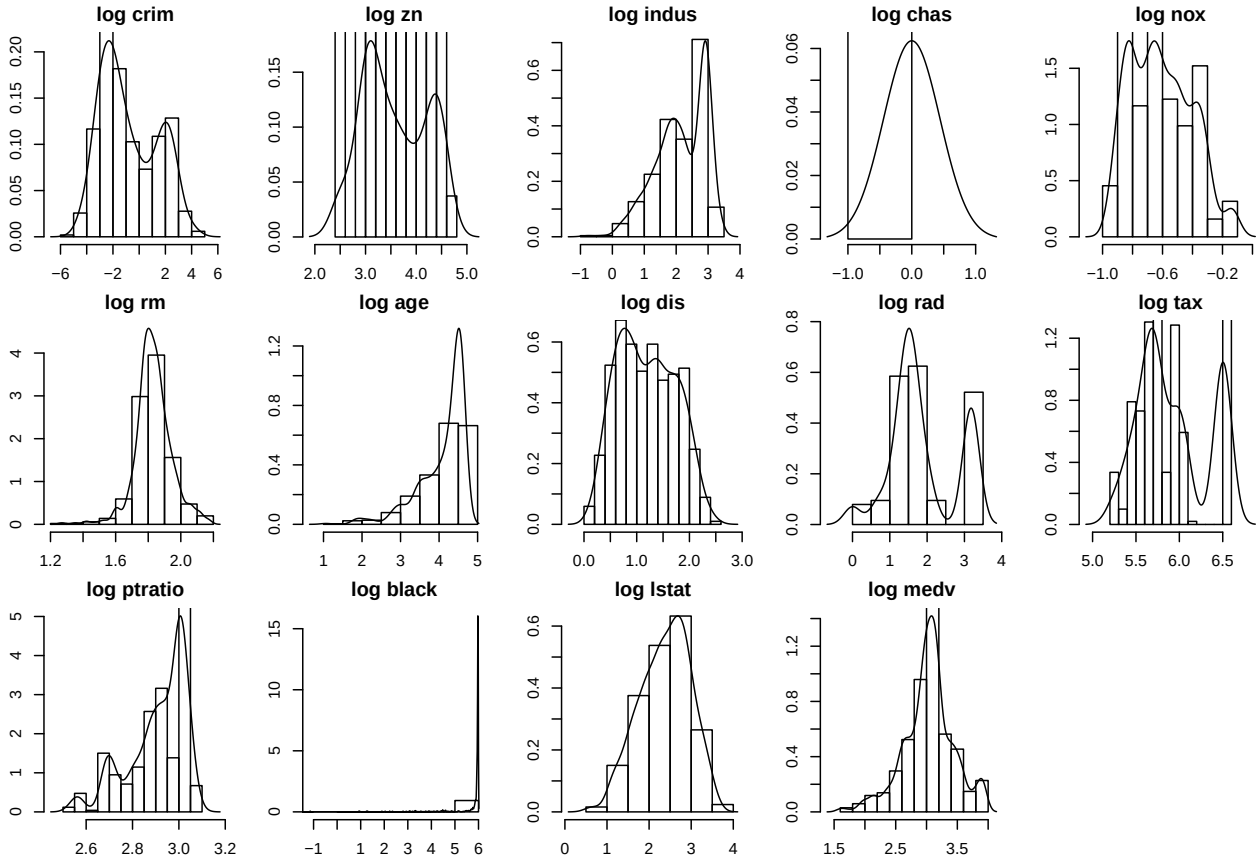


```
den=NULL
for(i in 1:ncol(Boston)){
  den <- density(Boston[,i])
  hist(Boston[,i], probability = T, xlim= range(den$x), ylim=range(den$y),
       main=colnames(Boston)[i])
  lines(den)}
```

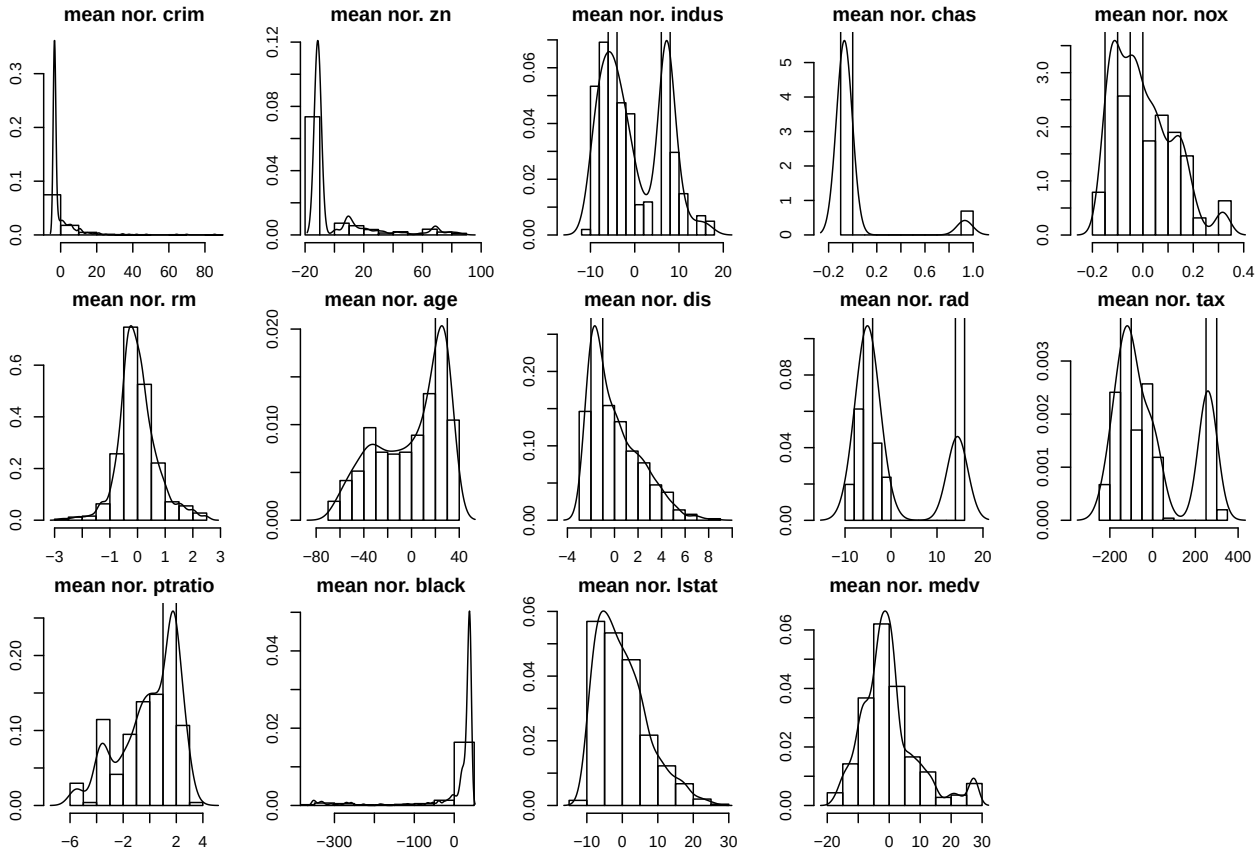


Variables `crim`, `black` and `medv` have a large number of outliers. Furthermore, some columns such as `crim`, `zn`, `chas` have a very skewed distribution and a heavy upper tail. Two normalization methods such as log transform and mean distance were applied to the data, variables visualized with histograms.

```
#log transformation
par(mfrow = c(3,5), mar=(c(2,2,2,2)))
den=NULL
for(i in 1:ncol(Boston)){
den <- density(log(Boston[,i]))
  hist(log(Boston[,i]), probability = T, xlim= range(den$x), ylim=range(den$y),
       main=paste("log",colnames(Boston)[i]))
  lines(den)}

# mean distance
norBoston <- apply(Boston, 2, function(x) x-mean(x, na.rm=T))
par(mfrow = c(3,5), mar=(c(2,2,2,2)))
```




```
den=NULL
for(i in 1:ncol(norBoston)){
  den <- density(norBoston[,i])
  hist(norBoston[,i], probability = T, xlim= range(den$x), ylim=range(den$y),
       main=paste("mean nor.", colnames(norBoston)[i]))
  lines(den)}
```



Upon log transformation, variables `crim, dis` and `lstat` showed a better symmetry and a more interpretable distribution. Therefore these variables were transformed before analyzing the data. Mean distance normalization did not provide better results, and was ignored. Furthermore, the description of the variables `zn, chas, rad` and `black` was not clear and interpretable therefore these variables were excluded from the data set.

```
boston <- subset(Boston, select=-c(zn, chas, rad, black))
boston[["crim"]] <- log(boston[["crim"]])
boston[["dis"]] <- log(boston[["dis"]])
boston[["lstat"]] <- log(boston[["lstat"]])

#summary of each varible
apply(boston,2,summary)
```

```
##           crim indus    nox    rm    age    dis   tax ptratio  lstat
## Min.    -5.0640  0.46 0.3850 3.561   2.90 0.1219 187.0   12.60 0.5481
## 1st Qu. -2.5000  5.19 0.4490 5.886  45.02 0.7420 279.0   17.40 1.9390
## Median  -1.3610  9.69 0.5380 6.208  77.50 1.1650 330.0   19.05 2.4300
## Mean    -0.7804 11.14 0.5547 6.285  68.57 1.1880 408.2   18.46 2.3710
## 3rd Qu.  1.3020 18.10 0.6240 6.624  94.07 1.6460 666.0   20.20 2.8310
## Max.     4.4880 27.74 0.8710 8.780 100.00 2.4950 711.0   22.00 3.6370
##           medv
## Min.     5.00
## 1st Qu. 17.02
## Median  21.20
## Mean    22.53
## 3rd Qu. 25.00
## Max.    50.00
```

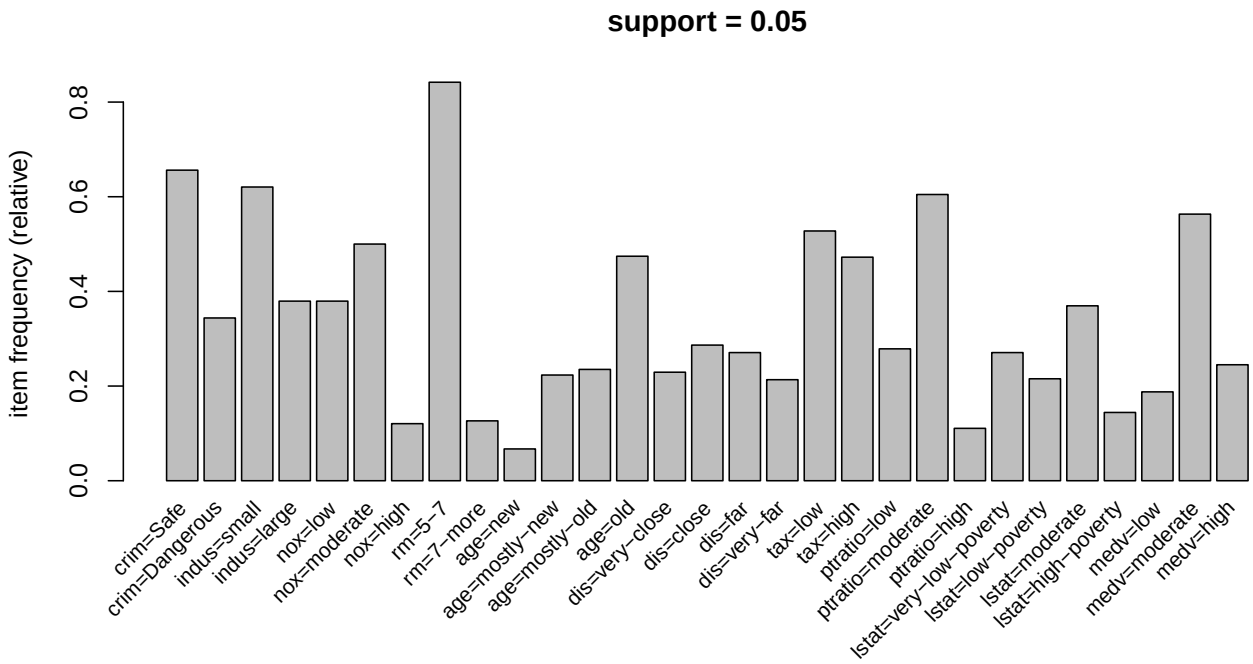
In order to transform the data into a binary incidence matrix, all continuous numeric variables should be transformed into ordered categories. To determine the cut off points for the discretization, histograms and summary results of the variables were considered. For example `crim` or criminality rate, show a bimodal distribution and the local minimal converges around 0, additionally according to the summary results, mean criminality is -0.7804 with a minimum of `-5.0640` and maximum of `4.4880` . Therefore `crim` was split into two groups and 0 was set as the cutoff point. Upon discretization, data was transform into a binary incidence matrix using the `arules` package.

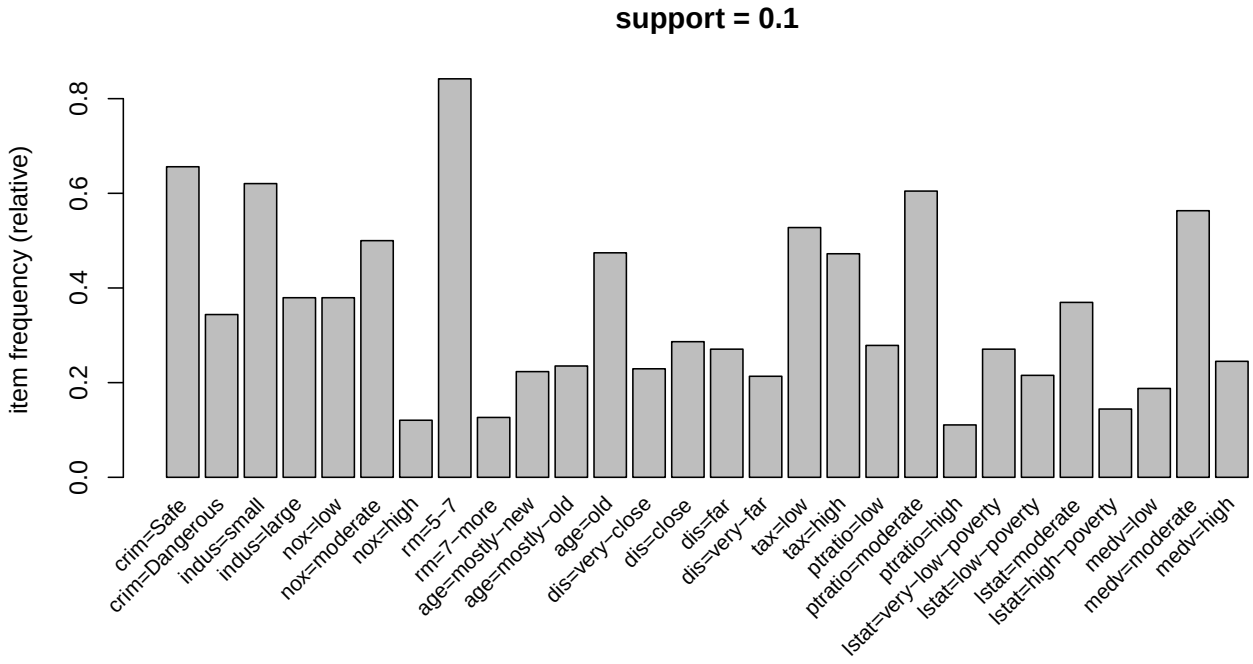
```
# discretization
boston[["crim"]] <- ordered(cut(boston[["crim"]], c(-6, 0, 5)) ,
                            labels=c("Safe", "Dangerous"))
boston[["indus"]] <- ordered(cut(boston[["indus"]], c(0,15,30)) ,
                            labels=c("small", "large"))
boston[["nox"]] <- ordered(cut(boston[["nox"]], c(0.3, 0.5, 0.7, 0.9)) ,
                            labels=c("low", "moderate", "high"))
boston[["rm"]] <- ordered(cut(boston[["rm"]], c(3, 5, 7, 9)) ,
                            labels=c("3-5", "5-7", "7-more"))
boston[["age"]] <- ordered(cut(boston[["age"]], c(2, 20, 50, 80, 100)) ,
                             labels=c("new","mostly-new","mostly-old", "old"))
boston[["dis"]] <- ordered(cut(boston[["dis"]], c(0, 0.7, 1.2, 1.7, 3)) ,
                           labels=c("very-close", "close", "far", "very-far"))
boston[["tax"]] <- ordered(cut(boston[["tax"]], c(180, 350, 750)) ,
                           labels=c("low", "high"))
boston[["ptratio"]] <- ordered(cut(boston[["ptratio"]], c(12.6,17.4,20.2,23)) ,
                           labels=c("low", "moderate", "high"))
boston[["lstat"]] <- ordered(cut(boston[["lstat"]], c(0, 2, 2.4, 3, 4)) ,
                           labels=c("very-low-poverty", "low-poverty","moderate","high-poverty"))
boston[["medv"]] <- ordered(cut(boston[["medv"]], c(5, 15, 25, 50)) ,
                             labels=c("low", "moderate", "high"))
#transformation
library(arules)
t_boston <- as(boston, "transactions")
#summary(t_boston)
```

(b) Visualize the data using the itemFrequencyPlot in the "arules" package. Apply the apriori algorithm

After transformation data was visualized using `itemFrequencyPlot` in the `arules` package for two different support values.

```
par(mfrow = c(2,1))
itemFrequencyPlot(t_boston, support = 0.05, cex.names= 0.8, main="support = 0.05")
itemFrequencyPlot(t_boston, support = 0.1, cex.names= 0.8, main="support = 0.1")
```






```
b_rules <- apriori(t_boston, parameter = list(support=0.05, confidence=0.6))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport support minlen maxlen
##         0.6    0.1    1 none FALSE            TRUE    0.05      1     10
##  target   ext
##   rules FALSE
```

```
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 25
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[30 item(s), 506 transaction(s)] done [0.00s].
## sorting and recoding items ... [29 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 7 8 9 done [0.00s].
## writing ... [14255 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

Support indicates how the recurrence/frequency of a certain transaction (in this case categories that appear together) will drive the analysis and whether low frequency transactions will be used for rule computation. As shown with two `itemFrequencyPlots`, selected support values did not heavily effect the categories that drive the analysis. Hence support `0.05` was preferred in order to be able compute more association rules. Two parameters were determined for the `apriori` function and `support` and `confidence` were set to `0.5` and `0.6` respectively. `0.6 confidance` indicates that only rules with more then 60% confidence will be computed.

(c) A student is interested is a low crime area as close to the city as possible (as measured by "dis"). What can you advise on this matter through the mining of association rules?

```
rulesCrimeLow <- subset(b_rules, subset = rhs %in% "crim=Safe" & lift>1.05)
inspect(head(sort(rulesCrimeLow, by = "support"), n=3))
```

```
##      lhs                      rhs         support   confidence lift
## 159  {indus=small}         => {crim=Safe} 0.6007905 0.9681529  1.475558
## 147  {tax=low}             => {crim=Safe} 0.5079051 0.9625468  1.467014
## 1348 {indus=small,rm=5-7} => {crim=Safe} 0.4920949 0.9651163  1.470930
```

The following rule was found {dis=far} => {crim=Safe} with `support = 0.2391304`, `confidence = 0.8832117` and `lift = 1.346100`, indicating that frequency of these two attributes occurring together are relatively high and with 88% confidence we can say that safe neighborhoods will likely be far from the city. According to the analysis it is very seldom to find a neighborhood that is safe and very close to the city. We can also see that the most of the safe towns have low industry, low taxes and dwellings with 5 to 7 rooms.

(d) A family is moving to the area, and has made schooling a priority. They want schools with low pupil-teacher ratios. What can you advise on this matter through the mining of association rules?

```
rulesPTLow <- subset(b_rules, subset = rhs %in% "ptratio=low" & lift>1.05)
inspect(head(sort(rulesPTLow, by = "support"), n=3))
```

```
##     lhs                           rhs           support    confidence
## 298 {rm=7-more,medv=high}    => {ptratio=low} 0.07312253 0.6271186
## 812 {nox=moderate,medv=high} => {ptratio=low} 0.07114625 0.7826087
## 318 {indus=small,rm=7-more}  => {ptratio=low} 0.06521739 0.6000000
##     lift
## 298 2.250511
## 812 2.808511
## 318 2.153191
```

Pupil-teacher ratio are contained in `ptratio` column. The results were sorted according to support and it can be seen that support values are relatively low, indicating that low pupil-teacher ratio is does not occur frequently. Additionally, neighborhoods that have low pupil-teacher ratio will likely have houses with a high median value and 7 or more rooms.