The following exercieses are from the following resource from page 60 to 63.

resource: [Friendly] Discrete Data Analysis with R: visualization and modeling techniques for categorical and count data, Michael Friendly, et al, ISBN: 978-1-4987-2583-5, CRC Press, 2015. [Available online at Seneca library]

Exercise 2.1 The packages vcd and vcdExtra contain many data sets with some examples of analysis and graphical display. The goal of this exercise is to familiarize yourself with these resources. You can get a brief summary of these using the function datasets() from vcdExtra. Use the following to get a list of these with some characteristics and titles.

```r
library(vcd)
```

```
## Loading required package: grid
```

```r
library(vcdExtra)
```

```
## Loading required package: gnm
```

```r
ds <- datasets(package = c("vcd", "vcdExtra"))
str(ds, vec.len = 2)
```

```
## 'data.frame':    78 obs. of  5 variables:
##  $ Package: chr  "vcd" "vcd" ...
##  $ Item   : chr  "Arthritis" "Baseball" ...
##  $ class  : chr  "data.frame" "data.frame" ...
##  $ dim    : chr  "84x5" "322x25" ...
##  $ Title  : chr  "Arthritis Treatment Data" "Baseball Data" ...
```

(a) How many data sets are there altogether? How many are there in each package?

```r
combine <- datasets(package = c("vcd", "vcdExtra"))

vcd_dataset <- datasets(package = "vcd")
vcdExtra_dataset <- datasets(package = "vcdExtra")

# total number of datasets altogehter
cat("Total number of datasets altogether:", nrow(combine), "\n")
```

```
## Total number of datasets altogether: 78
```

```r
# number of datasets in each packages
cat("Number of datasets in vcd package:", nrow(vcd_dataset), "\n")
```

```
## Number of datasets in vcd package: 33
```

```r
cat("Number of datasets in vcdExtra package:", nrow(vcdExtra_dataset), "\n")
```

```
## Number of datasets in vcdExtra package: 45
```

(b) Make a tabular display of the frequencies by Package and class.

```r
table(combine$Package, combine$class)
```

```
##
##             array data.frame table
##   vcd           1         17    15
##   vcdExtra      5         24    16
```

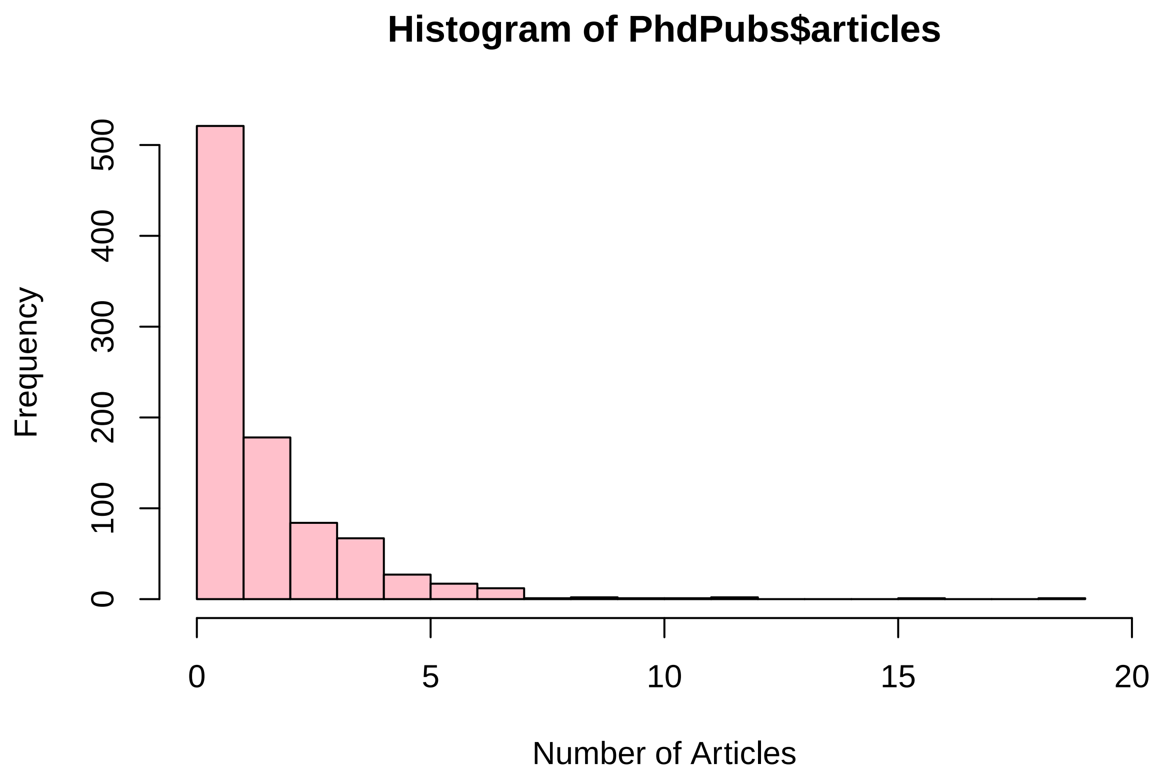
(c) Choose one or two data sets from this list, and examine their help files (e.g., help(Arthritis) or ?Arthritis). You can use, e.g., example(Arthritis) to run the R code for a given example.

```r
help(PhdPubs)
```

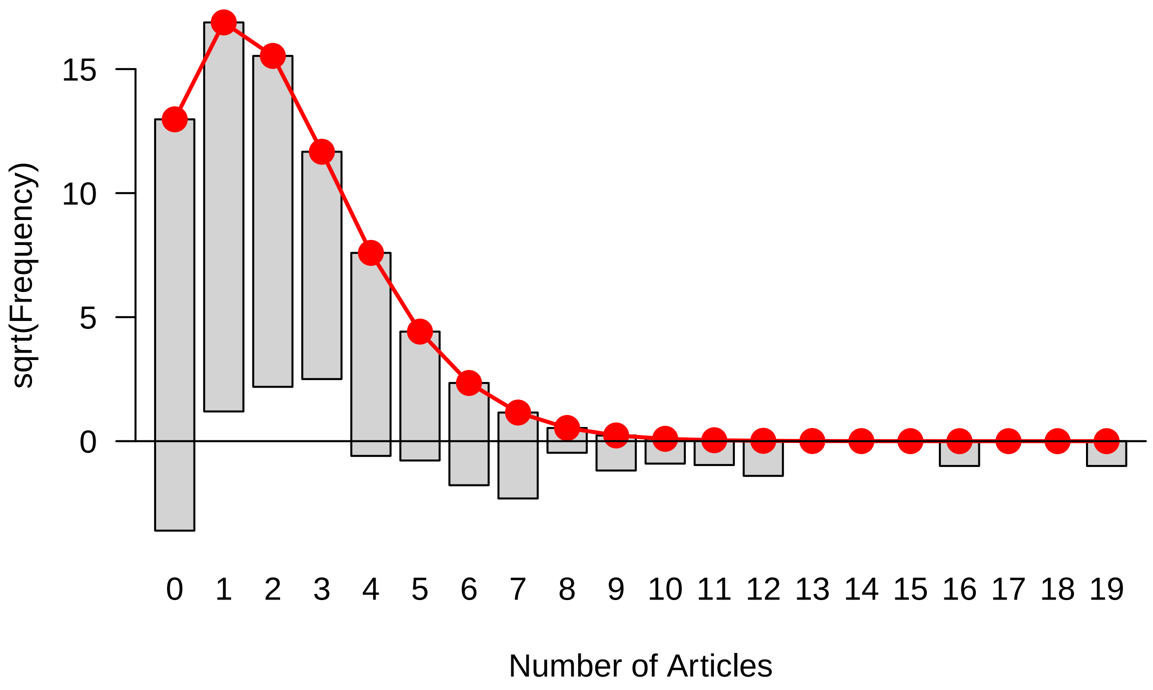
```
## starting httpd help server ... done
```

```r
example(PhdPubs)
```

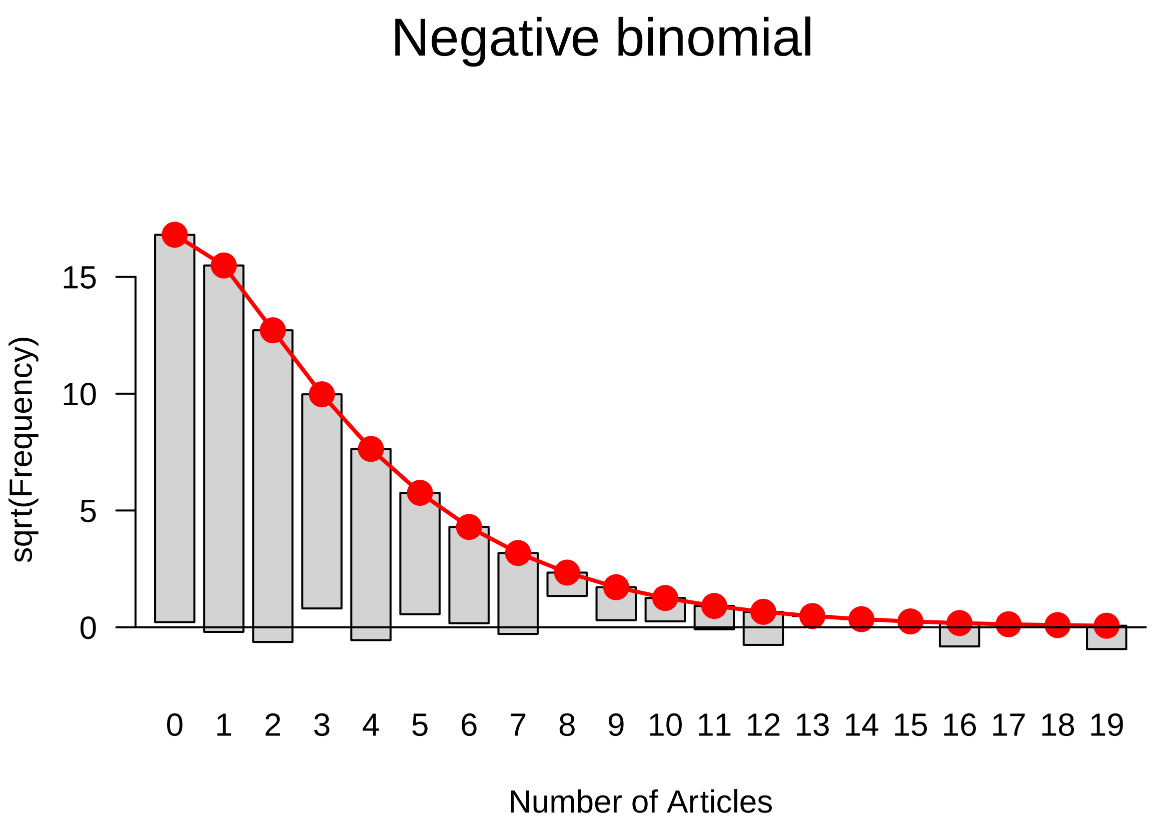
```
##
## PhdPbs> data(PhdPubs)
##
## PhdPbs> # very uninformative
## PhdPbs> hist(PhdPubs$articles,
## PhdPbs+      breaks=0:19, col="pink", xlim=c(0,20),
## PhdPbs+      xlab="Number of Articles")
```



```
##
## PhdPbs> library(vcd)
##
## PhdPbs> rootogram(goodfit(PhdPubs$articles), xlab="Number of Articles")
```



```
##
## PhdPbs> # compare with negative binomial
## PhdPbs> rootogram(goodfit(PhdPubs$articles, type="nbinomial"),
## PhdPbs+  xlab="Number of Articles", main="Negative binomial")
```



Exercise 2.2 For each of the following data sets in the vcdExtra package, identify which are response variable(s) and which are explanatory. For factor variables, which are unordered (nominal) and which should be treated as ordered? Write a sentence or two describing substantitive questions of interest for analysis of the data. (Hint: use data(foo, package="vcdExtra") to load, and str(foo), help(foo) to examine data set foo.)

(a) Abortion opinion data: Abortion **My Answer:** based on following information and also help() function, we can see that Sex and Status are explainatory unordered and nominal variables, and Support_abortion is the response.

```r
# using data() in order to show a dataset that come with a specific package.
# here Abortion dataset that comes with vcdExtra packages
data(Abortion, package="vcdExtra")
# display the structure, type and information about Abortion
str(Abortion)
```

```
##  'table' num [1:2, 1:2, 1:2] 171 152 138 167 79 148 112 133
##  - attr(*, "dimnames")=List of 3
##   ..$ Sex             : chr [1:2] "Female" "Male"
##   ..$ Status          : chr [1:2] "Lo" "Hi"
##   ..$ Support_Abortion: chr [1:2] "Yes" "No"
```

```r
help(Abortion)
```

(b) Caesarian Births: Caesar **My Answer:** based on following information and also help() function, we can see that Risk, Antibiotics and Planned are explanatory unordered and nominal variables, and Infection is the response.

```
# using data() in order to show a dataset that come with a specific package.
data(Caesar, package="vcdExtra")
# display the structure, type and information about dataset
str(Caesar)
```

```
##  'table' num [1:3, 1:2, 1:2, 1:2] 0 1 17 0 1 1 11 17 30 4 ...
##  - attr(*, "dimnames")=List of 4
##   ..$ Infection  : chr [1:3] "Type 1" "Type 2" "None"
##   ..$ Risk       : chr [1:2] "Yes" "No"
##   ..$ Antibiotics: chr [1:2] "Yes" "No"
##   ..$ Planned    : chr [1:2] "Yes" "No"
```

```
help(Caesar)
```

(c) Dayton Survey: DaytonSurvey **My Answer:** based on following information and also help() function, we can see that sex and race are unordered nominal explanatory variables and cigarette, alcohol, and marijuana are the response variables.

```
# using data() in order to show a dataset that come with a specific package.
data(DaytonSurvey, package="vcdExtra")
# display the structure, type and information about dataset
str(DaytonSurvey)
```

```
## 'data.frame':    32 obs. of  6 variables:
##  $ cigarette: Factor w/ 2 levels "Yes","No": 1 2 1 2 1 2 1 2 1 2 ...
##  $ alcohol  : Factor w/ 2 levels "Yes","No": 1 1 2 2 1 1 2 2 1 1 ...
##  $ marijuana: Factor w/ 2 levels "Yes","No": 1 1 1 1 2 2 2 2 1 1 ...
##  $ sex      : Factor w/ 2 levels "female","male": 1 1 1 1 1 1 1 1 2 2 ...
##  $ race     : Factor w/ 2 levels "white","other": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Freq     : num  405 13 1 1 268 218 17 117 453 28 ...
```

```
help(DaytonSurvey)
```

(d) Minnesota High School Graduates: Hoyt **My Answer:** based on following information and also help() function, we can see that “Rank,” “Occupation,” and “Sex” are unordered nominal explanatory variables and “Rank” and “Occupation” are ordered ordinal variables, and “Status” is the response variable.

```
# using data() in order to show a dataset that come with a specific package.
data(Hoyt, package="vcdExtra")
# display the structure, type and information about dataset
str(Hoyt)
```

```
##  'table' num [1:4, 1:3, 1:7, 1:2] 87 3 17 105 216 4 14 118 256 2 ...
##  - attr(*, "dimnames")=List of 4
##   ..$ Status    : chr [1:4] "College" "School" "Job" "Other"
##   ..$ Rank      : chr [1:3] "Low" "Middle" "High"
##   ..$ Occupation: chr [1:7] "1" "2" "3" "4" ...
##   ..$ Sex       : chr [1:2] "Male" "Female"
```

```r
help(Hoyt)
```

Exercise 2.3 The data set UCBAdmissions is a 3-way table of frequencies classified by Admit, Gender, and Dept.

(a) Find the total number of cases contained in this table.

```r
# total number of cases
cat("Total number of cases in UCBAdmissions dataset is:",
    sum(UCBAdmissions), "\n")
```

```
## Total number of cases in UCBAdmissions dataset is: 4526
```

(b) For each department, find the total number of applicants.

```r
cat("Total number of applicants in each departments of
    UCBAdmissions dataset are:\n")
```

```
## Total number of applicants in each departments of
##     UCBAdmissions dataset are:
```

```r
margin.table(UCBAdmissions,3)
```

```
## Dept
##   A   B   C   D   E   F
## 933 585 918 792 584 714
```

(c) For each department, find the overall proportion of applicants who were admitted.

```r
cat("Overall proportion of applicants who were admitted in each departments
    of UCBAdmissions dataset are:\n")
```

```
## Overall proportion of applicants who were admitted in each departments
##     of UCBAdmissions dataset are:
```

```r
datafram <-as.data.frame(UCBAdmissions)
department_frequency <- xtabs(Freq~Dept + Admit, data = datafram)
prop.table(department_frequency)
```

```
##     Admit
## Dept   Admitted   Rejected
##    A 0.13278833 0.07335395
##    B 0.08174989 0.04750331
##    C 0.07114450 0.13168361
##    D 0.05943438 0.11555457
##    E 0.03247901 0.09655325
##    F 0.01016350 0.14759169
```

(d) Construct a tabular display of department (rows) and gender (columns), showing the proportion of applicants in each cell who were admitted relative to the total applicants in that cell.

```r
freq_department_gender <- ftable(Gender ~ Admit + Dept, data = UCBAdmissions)
proportion_department_gender <- prop.table(freq_department_gender)
proportion_department_gender
```

```
##                Gender        Male      Female
## Admit    Dept
## Admitted A            0.113124171 0.019664163
##          B            0.077993814 0.003756076
##          C            0.026513478 0.044631021
##          D            0.030490499 0.028943880
##          E            0.011710119 0.020768891
##          F            0.004860804 0.005302696
## Rejected A            0.069155988 0.004197967
##          B            0.045735749 0.001767565
##          C            0.045293858 0.086389748
##          D            0.061643836 0.053910738
##          E            0.030490499 0.066062749
##          F            0.077551922 0.070039770
```

Exercise 2.4 The data set DanishWelfare in vcd gives a 4-way, 3 _ 4 _ 3 _ 5 table as a data frame in frequency form, containing the variable Freq and four factors, Alcohol, Income, Status, and Urban. The variable Alcohol can be considered as the response variable, and the others as possible predictors.

(a) Find the total number of cases represented in this table.

```r
cat("Total number of cases represented in DanishWelfare is:",
    sum(DanishWelfare$Freq), "\n")
```

```
## Total number of cases represented in DanishWelfare is: 5144
```

(b) In this form, the variables Alcohol and Income should arguably be considered ordered factors. Change them to make them ordered.

```r
# showing the levels of the "Alcohol" variable in the DanishWelfare dataset.
levels(DanishWelfare$Alcohol)
```

```
## [1] "<1"  "1-2" ">2"
```

```r
# Converting the "Alcohol" variable in the DanishWelfare
# dataset to an ordered factor.
DanishWelfare$Alcohol <- as.ordered(DanishWelfare$Alcohol)
# Converting the "Income" variable in the DanishWelfare
# dataset to an ordered factor.
DanishWelfare$Income <- as.ordered(DanishWelfare$Income)
# showing the structure of the DanishWelfare dataset in order
# to get information about its variables and their types.
str(DanishWelfare)
```

```
## 'data.frame':    180 obs. of  5 variables:
##  $ Freq   : num  1 4 1 8 6 14 8 41 100 175 ...
##  $ Alcohol: Ord.factor w/ 3 levels "<1"<"1-2"<">2": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Income : Ord.factor w/ 4 levels "0-50"<"50-100"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Status : Factor w/ 3 levels "Widow","Married",..: 1 1 1 1 1 2 2 2 2 2 ...
##  $ Urban  : Factor w/ 5 levels "Copenhagen","SubCopenhagen",..: 1 2 3 4 5 1 2 3 4 5 ...
```

(c) Convert this data frame to table form, DanishWelfare.tab, a 4-way array containing the frequencies with appropriate variable names and level names.

```r
# creating a cross table named DanishWelfare.tab,and then
# summarizing the frequencies of combinations of variables
# in the DanishWelfare dataset with variable names and level names.
DanishWelfare.tab <- xtabs(Freq ~ ., data = DanishWelfare)

# showing the structure of the DanishWelfare.tab table in order to get
# information about its dimensions and the frequencies
str(DanishWelfare.tab)
```

```
##  'xtabs' num [1:3, 1:4, 1:3, 1:5] 1 3 2 8 1 3 2 5 2 42 ...
##  - attr(*, "dimnames")=List of 4
##   ..$ Alcohol: chr [1:3] "<1" "1-2" ">2"
##   ..$ Income : chr [1:4] "0-50" "50-100" "100-150" ">150"
##   ..$ Status : chr [1:3] "Widow" "Married" "Unmarried"
##   ..$ Urban  : chr [1:5] "Copenhagen" "SubCopenhagen" "LargeCity" "City" ...
##  - attr(*, "call")= language xtabs(formula = Freq ~ ., data = DanishWelfare)
```

(d) The variable Urban has 5 categories. Find the total frequencies in each of these. How would you collapse the table to have only two categories, City, Non-city? **My Answer:** I used the collapse.table function to reduce the number of categories to just two. This function collapses the 'Urban' variable into two distinct categories: 'City' and 'NonCity.' Specifically, I reassigned the categories 'Copenhagen,' 'LargeCity,' and 'City' to 'City,' while 'SubCopenhagen' and 'Country' were reassigned to 'NonCity.' The vector Urban=c("City", "NonCity", "City", "City", "NonCity") indicates the mapping of the original categories to the new ones. This means that the first occurrence of 'Urban' being 'City' in the original data will be mapped to 'City' in the collapsed data, and so forth.

```r
# calculating the total population for each 5 category for the "Urban" variable
pop_urb <- margin.table(DanishWelfare.tab, 4)
pop_urb
```

```
## Urban
##    Copenhagen SubCopenhagen     LargeCity          City       Country 
##           552           614           594          1765          1619
```

```r
# dividing the "Urban" variable to two categories of "City"
# and "NonCity" instead of 5 categories
city_nonCity <- vcdExtra::collapse.table(
  DanishWelfare.tab, Urban=c("City", "NonCity", "City", "City", "NonCity"))

# showing the first few rows of the flattened table obtained after
# collapsing the categories in order to have a simplified view for
# analysis or visualization.
head(ftable(city_nonCity))
```

```
##
##                                    "Urban" "City" "NonCity"
##  "Alcohol" "Income"  "Status"
##  "<1"      "0-50"    "Widow"                    10        10
##                      "Married"                 155       183
##                      "Unmarried"                14        10
##            "50-100"  "Widow"                    29         7
##                      "Married"                 338       306
##                      "Unmarried"                36        32
```

(e) Use structable() or ftable() to produce a pleasing flattened display of the frequencies in the 4-way table. Choose the variables used as row and column variables to make it easier to compare levels of Alcohol across the other factors.

```r
structable(xtabs(Freq ~., data = DanishWelfare))
```

```
##                   Income       0-50                                               50-100
##                   Urban   Copenhagen SubCopenhagen LargeCity City Country Copenhagen SubCopenhagen La
## Alcohol Status
## <1      Widow                      1             4         1    8       6          8             2
##         Married                   14             8        41  100     175         42            51
##         Unmarried                  6             1         2    6       9          7             5
## 1-2     Widow                      3             0         1    4       2          1             1
##         Married                   15             7        15   25      48         39            59
##         Unmarried                  2             3         9    9       7         12             3
## >2      Widow                      2             0         2    1       0          3             0
##         Married                    1             2         2    7       7         14            21
##         Unmarried                  3             0         1    5       1          2             0
```

Exercise 2.5 The data set UKSoccer in vcd gives the distributions of number of goals scored by the 20 teams in the 1995/96 season of the Premier League of the UK Football Association.

```r
data("UKSoccer", package = "vcd")
ftable(UKSoccer)
```

```
##      Away  0  1  2  3  4
## Home
## 0         27 29 10  8  2
## 1         59 53 14 12  4
## 2         28 32 14 12  4
## 3         19 14  7  4  1
## 4          7  8 10  2  0
```

This two-way table classifies all 20 _ 19 = 380 games by the joint outcome (Home, Away), the number of goals scored by the Home and Away teams. The value 4 in this table actually represents 4 or more goals.

(a) Verify that the total number of games represented in this table is 380.

```r
cat("Total number of games is:", sum(UKSoccer), "\n")
```

```
## Total number of games is: 380
```

```r
ftable(UKSoccer)
```

```
##      Away  0  1  2  3  4
## Home
## 0         27 29 10  8  2
## 1         59 53 14 12  4
## 2         28 32 14 12  4
## 3         19 14  7  4  1
## 4          7  8 10  2  0
```

(b) Find the marginal total of the number of goals scored by each of the home and away teams.

```r
cat("the marginal total of the number of goals scored by each
    of the home and away teams:\n\n number 1 in the
    [prop.table(UKSoccer,1)] means proportions based on each
    row (e.g., for each home team).\n\n")
```

```
## the marginal total of the number of goals scored by each
##     of the home and away teams:
##
##  number 1 in the
##     [prop.table(UKSoccer,1)] means proportions based on each
##     row (e.g., for each home team).
```

```r
prop.table(UKSoccer,1)
```

```
##     Away
## Home          0          1          2          3          4
##    0 0.35526316 0.38157895 0.13157895 0.10526316 0.02631579
##    1 0.41549296 0.37323944 0.09859155 0.08450704 0.02816901
##    2 0.31111111 0.35555556 0.15555556 0.13333333 0.04444444
##    3 0.42222222 0.31111111 0.15555556 0.08888889 0.02222222
##    4 0.25925926 0.29629630 0.37037037 0.07407407 0.00000000
```

```r
cat("\n number 2 in the [prop.table(UKSoccer,2)] means
    proportions based on each column (e.g., for each away team).\n\n")
```

```
##
##  number 2 in the [prop.table(UKSoccer,2)] means
##     proportions based on each column (e.g., for each away team).
```

```r
prop.table(UKSoccer,2)
```

```
##     Away
## Home          0          1          2          3          4
##    0 0.19285714 0.21323529 0.18181818 0.21052632 0.18181818
##    1 0.42142857 0.38970588 0.25454545 0.31578947 0.36363636
##    2 0.20000000 0.23529412 0.25454545 0.31578947 0.36363636
##    3 0.13571429 0.10294118 0.12727273 0.10526316 0.09090909
##    4 0.05000000 0.05882353 0.18181818 0.05263158 0.00000000
```

(c) Express each of the marginal totals as proportions. **My Answer:** The code prop.table(margin.table(UKSoccer,1)) calculates the marginal totals along margin 1, corresponding to the row totals, which signify the total number of goals scored by home teams for each possible goal count (0, 1, 2, 3, 4). The subsequent use of prop.table(…) transforms these raw counts into proportions, expressing the distribution of goals scored by home teams as a percentage of the overall home team goals. The resulting output displays the proportions for each goal count (0, 1, 2, 3, 4) for home teams, illustrating the percentage representation of each goal category. For instance, it highlights that the proportion of home teams scoring 0 goals is 20%, while the proportion of home teams scoring 1 goal is 37.37%. **My Answer:** The code prop.table(margin.table(UKSoccer,2)) computes the proportions of goals scored by away teams for each possible goal count (0, 1, 2, 3, 4) in the UKSoccer dataset. First, margin.table(UKSoccer,2) calculates the column totals, representing the total number of goals scored by away teams for each goal count. Subsequently, prop.table transforms these raw counts into proportions, expressing the distribution of goals by away teams as a percentage of the overall away team goals. The resulting output displays the proportions for each goal count, such as the proportion of away teams scoring 0 goals (36.84%) and the proportion scoring 1 goal (35.79%). This analysis provides insights into the relative contribution of each goal count to the total goals scored by away teams.

```
prop.table(margin.table(UKSoccer,1))
```

```
## Home
##          0          1          2          3          4
## 0.20000000 0.37368421 0.23684211 0.11842105 0.07105263
```

```
prop.table(margin.table(UKSoccer,2))
```

```
## Away
##          0          1          2          3          4
## 0.36842105 0.35789474 0.14473684 0.10000000 0.02894737
```

(d) Comment on the distribution of the numbers of home-team and away-team goals. Is there any evidence that home teams score more goals on average? **My Answer:** Comment:For the distribution of home-team goals, the proportions reveal that the majority of home teams score 0 or 1 goal, with 20% scoring 0 goals and 37.37% scoring 1 goal. The proportions decrease for higher goal counts (2, 3, 4), indicating a skewed distribution towards lower goal counts. **My Answer:** On the other hand, the distribution of away-team goals shows a relatively even spread across different goal counts. The proportions for away teams scoring 0, 1, 2, 3, and 4 goals are 36.84%, 35.79%, 14.47%, 10%, and 2.89%, respectively. This suggests a more balanced distribution compared to home teams.

```
cat("\nEvidence that home teams score more goals on average is that the
    average number of goals weighted with their marginal frequencies for home
    teams is: ", weighted.mean(0:4, w=margin.table(UKSoccer,1)), ",
    but for away teams, this average number is:
    ", weighted.mean(0:4, w=margin.table(UKSoccer,2)), ".\n\n")
```

```
##
## Evidence that home teams score more goals on average is that the
##     average number of goals weighted with their marginal frequencies for home
##     teams is:  1.486842 ,
##     but for away teams, this average number is:
##      1.063158 .
```

Exercise 2.6 The one-way frequency table Saxony in vcd records the frequencies of families with 0, 1, 2, : : : 12 male children, among 6115 families with 12 children. This data set is used extensively in Chapter 3.

```r
data("Saxony", package = "vcd")
Saxony
```

```
## nMales
##    0    1    2    3    4    5    6    7    8    9   10   11   12
##    3   24  104  286  670 1033 1343 1112  829  478  181   45    7
```

Another data set, Geissler, in the vcdExtra package, gives the complete tabulation of all combinations of boys and girls in families with a given total number of children (size). The task here is to create an equivalent table, Saxony12 from the Geissler data.

```r
data("Geissler", package = "vcdExtra")
str(Geissler)
```

```
## 'data.frame':    90 obs. of  4 variables:
##  $ boys : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ girls: num  1 2 3 4 5 6 7 8 9 10 ...
##  $ size : num  1 2 3 4 5 6 7 8 9 10 ...
##  $ Freq : int  108719 42860 17395 7004 2839 1096 436 161 66 30 ...
```

(a) Use subset() to create a data frame, sax12 containing the Geissler observations in families with size==12.

```r
# Create a subset of the 'Geissler' dataframe where the 'size' column
# is equal to 12 (observations in families with total 12 children.
sax12 <- subset(Geissler, Geissler$size == 12)

# Display the resulting subset 'sax12'.
sax12
```

```
##    boys girls size Freq
## 12    0    12   12    3
## 24    1    11   12   24
## 35    2    10   12  104
## 45    3     9   12  286
## 54    4     8   12  670
## 62    5     7   12 1033
## 69    6     6   12 1343
## 75    7     5   12 1112
## 80    8     4   12  829
## 84    9     3   12  478
## 87   10     2   12  181
## 89   11     1   12   45
## 90   12     0   12    7
```

(b) Select the columns for boys and Freq.

```r
# Subset the 'sax12' data frame and select only the columns 'boys' and 'Freq'
subsetted_data <- subset(sax12, select = c(boys, Freq))
subsetted_data
```

```
##    boys Freq
## 12    0    3
## 24    1   24
## 35    2  104
## 45    3  286
## 54    4  670
## 62    5 1033
## 69    6 1343
## 75    7 1112
## 80    8  829
## 84    9  478
## 87   10  181
## 89   11   45
## 90   12    7
```

(c) Use xtabs() with a formula, Freq ~ boys, to create the one-way table.

```r
xtabs(Freq ~ boys,data = sax12)
```

```
## boys
##    0    1    2    3    4    5    6    7    8    9   10   11   12 
##    3   24  104  286  670 1033 1343 1112  829  478  181   45    7
```

(d) Do the same steps again to create a one-way table, Saxony11, containing similar frequencies for families of size==11.

```r
Saxony11 <- subset(Geissler, Geissler$size==11)
xtabs(Freq ~ boys,data = Saxony11)
```

```
## boys
##    0    1    2    3    4    5    6    7    8    9   10   11 
##    8   72  275  837 1540 2161 2310 1801 1077  492   93   24
```

Exercise 2.7 Interactive coding of table factors: Some statistical and graphical methods for contingency tables are implemented only for two-way tables, but can be extended to 3+-way tables by recoding the factors to interactive combinations along the rows and/or columns, in a way similar to what ftable() and structable() do for printed displays. For the UCBAdmissions data, produce a two-way table object, UCB.tab2, that has the combinations of Admit and Gender as the rows, and Dept as its columns, to look like the result below:

```r
# Create a two-way table with Admit, Gender, and Dept
UCB.tab2 <- ftable(Dept~ Admit + Gender, data = UCBAdmissions)

# Display the resulting table
print(UCB.tab2)
```

```
##                  Dept   A   B   C   D   E   F
## Admit    Gender
## Admitted Male        512 353 120 138  53  22
##          Female       89  17 202 131  94  24
## Rejected Male        313 207 205 279 138 351
##          Female       19   8 391 244 299 317
```

(a) Try this the long way: convert UCBAdmissions to a data frame (as.data.frame()), manipulate the factors (e.g., interaction()), then convert back to a table (as.data.frame()).

```
# Convert UCBAdmissions to a data frame
UCBAdmissions_df <- as.data.frame(UCBAdmissions)

# Manipulate factors, for example, using interaction()
UCBAdmissions_df$Admit_Gender <- with(UCBAdmissions_df, interaction(Admit, Gender, sep = ":"))

# Create a subset of the data frame, selecting only the
# columns 'Dept', 'Admit_Gender', and 'Freq'
ucb_df_subset <- subset(UCBAdmissions_df, select = c("Dept", "Admit_Gender", "Freq"))

# Create a contingency table 'ucb_df_subset.tab2'
# using the 'xtabs' function,with frequencies 'Freq' modeled as a
# function of the interaction of 'Admit_Gender' and 'Dept'
ucb_df_subset.tab2 <- xtabs(Freq ~ Admit_Gender + Dept, data = ucb_df_subset)

# Display the resulting contingency table 'ucb_df_subset.tab2'
ucb_df_subset.tab2
```

```
##                  Dept
## Admit_Gender        A   B   C   D   E   F
##   Admitted:Male   512 353 120 138  53  22
##   Rejected:Male   313 207 205 279 138 351
##   Admitted:Female  89  17 202 131  94  24
##   Rejected:Female  19   8 391 244 299 317
```

(b) Try this the short way: both ftable() and structable() have as.matrix() methods that convert their result to a matrix.

```
ucb.tab2 <- as.matrix(structable(Dept ~ Admit + Gender, data = UCBAdmissions))
ucb.tab2
```

```
##                  Dept
## Admit_Gender        A   B   C   D   E   F
##   Admitted_Male   512 353 120 138  53  22
##   Admitted_Female  89  17 202 131  94  24
##   Rejected_Male   313 207 205 279 138 351
##   Rejected_Female  19   8 391 244 299 317
```

Exercise 2.8 The data set VisualAcuity in vcd gives a 4 __ 4 __ 2 table as a frequency data frame.

```r
data("VisualAcuity", package = "vcd")
str(VisualAcuity)
```

```
## 'data.frame':    32 obs. of  4 variables:
##  $ Freq  : num  1520 234 117 36 266 ...
##  $ right : Factor w/ 4 levels "1","2","3","4": 1 2 3 4 1 2 3 4 1 2 ...
##  $ left  : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 2 2 2 2 3 3 ...
##  $ gender: Factor w/ 2 levels "male","female": 2 2 2 2 2 2 2 2 2 2 ...
```

(a) From this, use xtabs() to create two 4 _ 4 frequency tables, one for each gender.

```r
data("VisualAcuity", package="vcd")
# Create a frequency table for males
male_freq_table <- xtabs(Freq ~ right + left,
                         data = subset(VisualAcuity, gender == "male"))

# Create a frequency table for females
female_freq_table <- xtabs(Freq ~ right + left,
                           data = subset(VisualAcuity, gender == "female"))

# Display the frequency tables
cat("Frequency Table for Males:\n\n")
```

```
## Frequency Table for Males:
```

```r
male_freq_table
```

```
##      left
## right   1   2   3   4
##     1 821 112  85  35
##     2 116 494 145  27
##     3  72 151 583  87
##     4  43  34 106 331
```

```r
cat("\n\nFrequency Table for Females:\n\n")
```

```
##
##
## Frequency Table for Females:
```

```r
female_freq_table
```

```
##      left
## right    1    2    3    4
##     1 1520  266  124   66
##     2  234 1512  432   78
##     3  117  362 1772  205
##     4   36   82  179  492
```

(b) Use structable() to create a nicely organized tabular display.

```r
structable(right ~ left + gender, data = xtabs(Freq ~ ., data = VisualAcuity))
```

```
##             right    1    2    3    4
## left gender
## 1    male          821  116   72   43
##      female       1520  234  117   36
## 2    male          112  494  151   34
##      female        266 1512  362   82
## 3    male           85  145  583  106
##      female        124  432 1772  179
## 4    male           35   27   87  331
##      female         66   78  205  492
```

(c) Use xtable() to create a LATEX or HTML table.

```r
library(xtable)
HTML_table <- xtable(female_freq_table)
print(HTML_table, type="html")
```

```
## <!-- html table generated in R 4.3.2 by xtable 1.8-4 package -->
## <!-- Thu Feb  8 00:03:22 2024 -->
## <table border=1>
## <tr> <th>  </th> <th> 1 </th> <th> 2 </th> <th> 3 </th> <th> 4 </th>  </tr>
##   <tr> <td align="right"> 1 </td> <td align="right"> 1520.00 </td> <td align="right"> 266.00 </td> <t
##   <tr> <td align="right"> 2 </td> <td align="right"> 234.00 </td> <td align="right"> 1512.00 </td> <t
##   <tr> <td align="right"> 3 </td> <td align="right"> 117.00 </td> <td align="right"> 362.00 </td> <td
##   <tr> <td align="right"> 4 </td> <td align="right"> 36.00 </td> <td align="right"> 82.00 </td> <td a
##    </table>
```