R for the Learned

Jim Bentley

June 13, 2012

1 Loading R Packages

While R itself is powerful, the use of packages makes it even more so. By using code generated by others and not having to reinvent the wheel, the user can save a lot of time.

Packages must first be installed from a CRAN or other source. Although periodic updating is suggested, once the R workspace is saved reinstallation is not necessary.

```
> ##install.packages can be used in place of the R-gui
> #install.packages("xtable")
> ## load a few helpful packageslibrary(lattice)
> library(Rcmdr)
> library(Hmisc)
> library(xtable)
> library(ggplot2)
> library(survival)
```

2 Data Entry

Data may be entered into R in a number of ways. Three commonly used methods will be discussed.

2.1 Manual Entry

Perhaps the easiest way to enter small datasets is to enter each variable indvidually and then combine them into a `data frame`. Using the data from BPS5 problem 4.9, this might look like:

```
> sex = c(rep("Female",12),rep("Male",7))
> mass = c(36.1, 54.6, 48.5, 42.0, 50.6, 42.0, 40.3, 33.1, 42.4,
+          34.5, 51.1, 41.2, 51.9, 46.9, 62, 62.9, 47.4, 48.7, 51.9)
> rate = c(995, 1425, 1396, 1418, 1502, 1256, 1189, 913, 1124, 1052,
+          1347, 1204, 1867, 1439, 1792, 1666, 1362, 1614, 1460)
> gender = c(rep(1,12),rep(2,7))
> bps5.4.9 = data.frame(sex, mass, rate, gender)
```

We can now check to see if the data frame has been created by entering

```
> ls()

[1] "bps5.4.9" "gender"   "mass"     "rate"     "sex"
```

Note that the listing also shows the individual variables that were used to create the data frame. These can be deleted by using rm().

```
> rm("sex", "mass", "rate", "gender")
> ls()

[1] "bps5.4.9"
```

The attributes of the data frame and some summary statistics can be computed using the `attributes` and `summary` functions.

```
> attributes(bps5.4.9)

$names
[1] "sex"    "mass"   "rate"   "gender"

$row.names
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19

$class
[1] "data.frame"

> summary(bps5.4.9)

     sex          mass            rate          gender
 Female:12   Min.   :33.10   Min.   : 913   Min.   :1.000
 Male  : 7   1st Qu.:41.60   1st Qu.:1196   1st Qu.:1.000
             Median :47.40   Median :1396   Median :1.000
             Mean   :46.74   Mean   :1370   Mean   :1.368
             3rd Qu.:51.50   3rd Qu.:1481   3rd Qu.:2.000
             Max.   :62.90   Max.   :1867   Max.   :2.000
```

Notice that while `sex` was treated as a categorical variable, `gender` was treated as if it was cardinal. R is smart in that it recognizes the difference between cardinal and categorical (which it calls "factor") variables. To make `gender` a factor variable we can enter

```
> bps5.4.9$gender = factor(bps5.4.9$gender,levels=c(1,2),
+                          labels=c("F","M"))
```

Using `summary` we can see that `gender` is treated as a factor, or categorical, variable.

```
> summary(bps5.4.9)

     sex          mass             rate      gender
 Female:12   Min.   :33.10   Min.   : 913   F:12
 Male  : 7   1st Qu.:41.60   1st Qu.:1196   M: 7
             Median :47.40   Median :1396
             Mean   :46.74   Mean   :1370
             3rd Qu.:51.50   3rd Qu.:1481
             Max.   :62.90   Max.   :1867
```

2.2 Using Rcmdr

The package `Rcmdr` allows us to import data created in a number of packages. While the Windows version of R will import Excel (.XLS) files, the Mac version of R does not. However, both versions will import SPSS transport files.

To use Rcmdr we first need to load the package. This can be accomplished using menus or by using the `library` function. Assuming that `Rcmdr` is installed we enter

```
> library(Rcmdr)
```

If everything is working correctly, the `Rcmdr` GUI interface should start. After selecting **Data** – **Import Data** – **from Excel, Access, or dBase data set**, R will ask us for a name for our data set. Enter something descriptive but easy to type (*e.g.* `HtWt`). Remeber that R is case sensitive.

Next, you will have to select the Excel file that contains your data. R will then ask which sheet in the Excel file you wish to import. Once you have selected a sheet, R will complete the import and the data set/frame will be created.

`Rcmdr` will indicate that the data frame has been created and selected by showing `Data set: HtWt` above the script window. You can now view the data by clicking on `View data set`.

Noting that the `Group` variable (which is really a sex variable) is coded as a numeric (1 or 2), we should probably recode it as a factor variable. `Rcmrd` makes this easy. Click on **Data** – **Manage variables in active data set** – `Convert numeric variables to factors`. Select the variable we wish to change — in this case `Group`. We will supply level names and use the same variable for the factor recoding. Click on **OK**. We are going to overwrite `Group` so click on **Yes**. In this case a 1 is a Male and a 2 is a Female. Once the level names have been entered, click on **OK**.

Clicking on **View data set** we see that the `Group` variable is now coded as Female and Male. R now recognizes `Group` as a factor/categorical variable.

Data that is stored in SPSS portable or save formats can be imported in a similar manner. The files that come with BPS5e are actually in the portable format so you can use the menus to create a new data frame.

2.3 Reading Comma Seperated Value (CSV) Files

R has a utility for reading comma seperated value (CSV) ascii files. These files can reside on the host machine or on a server. If the files are in standard CSV format,

either of

```
> HtWt = read.csv("c:/stat/ncssdata/htwt.csv")
> htwt = read.csv(
+ "http://newton.uor.edu/facultyfolder/jim_bentley/downloads/math111/htwt.csv")
```

will create a data frame that contains the NCSS Sample data set's height and weight data. Note the use of forward slashes instead of backslashes.

The group variable will be imported as a numeric. To help R function efficiently, it will need to be converted to a factor variable using one of the methods from above.

2.4 Saving and Loading Data Frames

Regardless of how they were created, data frames may be saved in R as part of the R workspace. The workspace contains all of the variables, data frames, and functions that you have defined. A workspace is a snapshot of your work to the point of the save.

To save a workspace click on **File** – **Save Workspace**. Select the folder to which you wish to save the file and a file name and then click on **Save**. Your workspace is now safely tucked away on your drive. This file can later be **Load**ed or you can open it by double clicking on the file.

History files store the commands that you used during your R session. These can be saved ans loaded in a manner similar to that of workspaces. These files are are text files and can be edited using Wordpad or something similar.

3 Graphics

R contains a number of predefined data frames. Some of these will be used in the examples that are presented below.

R supports a number of different approaches to generating graphics. We will look at standard R graphics, the lattice package, and graphics using the ggplot2 package.

3.1 Standard R Graphics

To use the standard graphics within R we do not need to load any additional packages. A simple scatterplot of the data from BPS5e problem 4.9 (Figure 1) can be created by entering

```
> plot(bps5.4.9$mass,bps5.4.9$rate,
+      xlab="Lean Body Mass (kilograms)",
+      ylab="Metabolic Rate (calories)")
```

A boxplot of the rate variable (Figure 2) can be generated using

```
> boxplot(bps5.4.9$rate, ylab="Metabolic Rate (calories)")
```

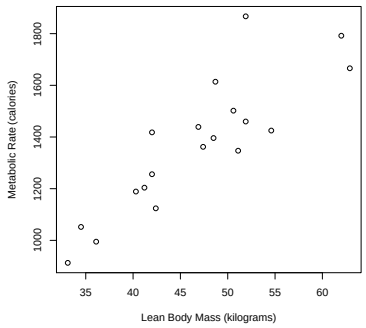

Figure 1: Plot of metabolic rate as a function of lean body mass for the data from BPS5 problem 4.9.

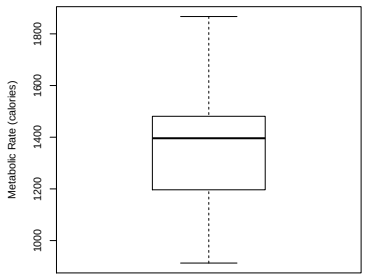

Figure 2: Boxplot of metabolic rate for the data from BPS5 problem 4.9.

A histogram of metabolic rate for the data from BPS5 problem 4.9 (Figure 3) can be generated using

```
> hist(bps5.4.9$rate, xlab="Metabolic Rate (calories)")
```

The corresponding stemplot for the rate data is given by entering

```
> print(stem(bps5.4.9$rate))

  The decimal point is 2 digit(s) to the right of the |

   8 | 1
  10 | 0529
  12 | 0656
  14 | 023460
  16 | 179
  18 | 7
```

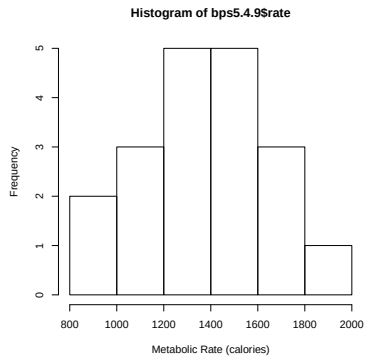

Figure 3: Histogram of the metabolic rates (colories) from BPS5e problem 4.9.

```
NULL
```

Since this generates a stemplot with too few stems, we may wish to expand the stems a bit. The following function call provides more stems—10 to be exact.

```
> print(stem(bps5.4.9$rate,2))

  The decimal point is 2 digit(s) to the right of the |

   9 | 1
  10 | 05
  11 | 29
  12 | 06
  13 | 56
  14 | 02346
  15 | 0
  16 | 17
  17 | 9
  18 | 7

NULL
```

Of course, it is possible to have too many stems as is shown in the following example.

```
> print(stem(bps5.4.9$rate,5))

  The decimal point is 2 digit(s) to the right of the |

   9 | 1
   9 |
  10 | 0
```

```
  10 | 5
  11 | 2
  11 | 9
  12 | 0
  12 | 6
  13 |
  13 | 56
  14 | 0234
  14 | 6
  15 | 0
  15 |
  16 | 1
  16 | 7
  17 |
  17 | 9
  18 |
  18 | 7

NULL
```

3.2 Lattice Graphics

Use of the `lattice` package requires that the package be loaded. Entering

```
> library(lattice)
```

accomplishes this.

A simple scatterplot of the data from BPS5e problem 4.9 (Figure 4) can be created by entering

```
> latticeplot = xyplot(rate~mass, data=bps5.4.9,
+                      xlab="Lean Body Mass (kilograms)",
+                      ylab="Metabolic Rate (calories)")
> print(latticeplot)
```

Comparison of sexes can be made by using conditioning (Figure 5).

```
> latticeplot = xyplot(rate~mass|sex, data=bps5.4.9,
+                      xlab="Lean Body Mass (kilograms)",
+                      ylab="Metabolic Rate (calories)")
> print(latticeplot)
```

or by the using different symbols for the two groups in overlayed plots (Figure 6).

```
> latticeplot = xyplot(rate~mass, group=sex,
+                      pch=c(1,3), data=bps5.4.9,
+                      xlab="Lean Body Mass (kilograms)",
+                      ylab="Metabolic Rate (calories)")
> print(latticeplot)
```

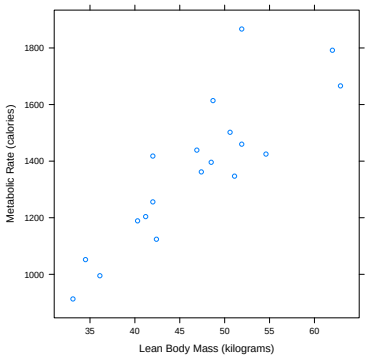

Figure 4: Plot of metabolic rate as a function of lean body mass for the data from BPS5 problem 4.9.

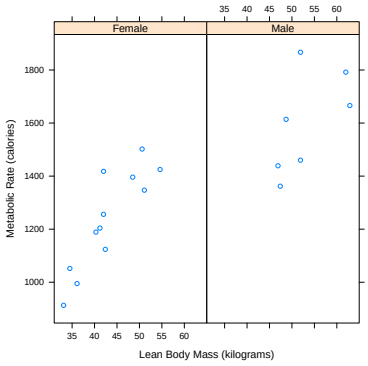

Figure 5: Plot of metabolic rate as a function of lean body mass while controlling for sex for the data from BPS5 problem 4.9.

A boxplot of the rate variable (Figure 7) can be generated using

```
> latticeplot = bwplot(~rate, data=bps5.4.9,
+                      xlab="Metabolic Rate (calories)")
> print(latticeplot)
```

A boxplot of the rate variable comparing sexes (Figure 8) can be generated using

```
> latticeplot = bwplot(sex~rate, data=bps5.4.9,
+                      ylab="Sex", xlab="Metabolic Rate (calories)")
> print(latticeplot)
```

The `lattice` package includes a few sample data frames. One of these is the `singer` data frame that contains information on various characteristics of some group of singers.

We can create a histogram of the heights of the singers (Figure 9) using

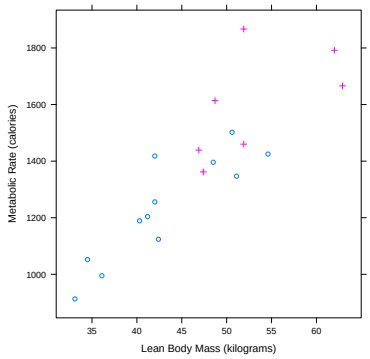

Figure 6: Plot of metabolic rate as a function of lean body mass while controlling for sex for the data from BPS5 problem 4.9.

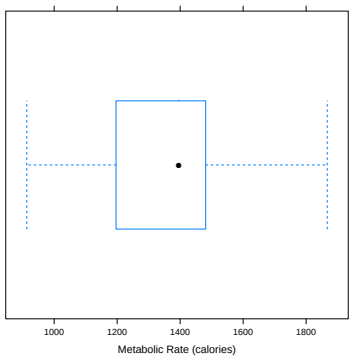

Figure 7: Boxplot of metabolic rate for the data from BPS5 problem 4.9.

```
> latticeplot = histogram(~height, data=singer)
> print(latticeplot)
```

We can gain additional information by controlling for voice part when creating a histogram of the heights of the singers (Figure 10) using

```
> latticeplot = histogram(~height|voice.part, data=singer)
> print(latticeplot)
```

Similarly, we can look at the distribution of the heights of the singers using density plots. Again, we can gain additional information by controlling for voice part (Figure 11).

```
> latticeplot = densityplot(~height|voice.part,data=singer)
> print(latticeplot)
```

One of the nice things about R is that its use of objects means that it is smart about data types. R knows the difference between cardinal (numerical) and categorical

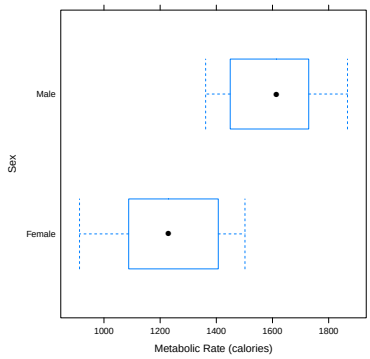

Figure 8: Boxplot of metabolic rate controlling for sex for the data from BPS5 problem 4.9.

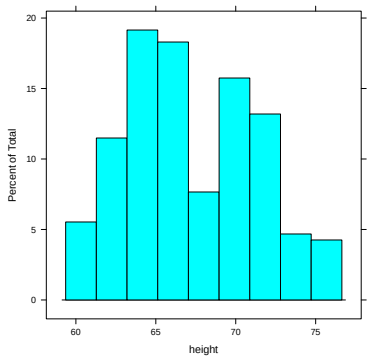

Figure 9: Histogram of the heights of singers in the `singer` data frame.

(factor) data. The `histogram` function from the `lattice` package will revert to a bargraph when asked to plot a factor variable. Figure 12 shows how this works for the `voice.part` variable.

```
> latticeplot = histogram(~voice.part,data=singer)
> print(latticeplot)
```

Figure 13 is the plot that made the whole idea of trellised graphics famous. The barley data that is presented had been analyzed for years by both the investigators and students. It was not until trellised graphics came along that it was recognized that one of the sites appears to have had its year data swapped.

```
> latticeplot = dotplot(variety ~ yield | site, data = barley,
+                       groups = year, pch=c(1,3),
+                       key = simpleKey(levels(barley$year),
+                       space = "right"),
+                       xlab = "Barley Yield (bushels/acre) ",
```

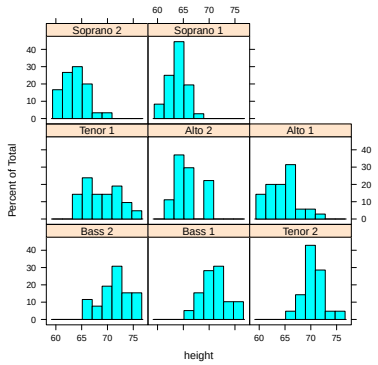

Figure 10: Histogram of the heights of singers in the `singer` data frame controlling for voice part.

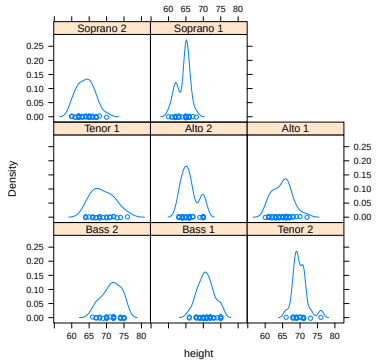

Figure 11: Density plot of the heights of singers in the `singer` data frame controlling for voice part.

```
+                          aspect=0.5, layout = c(1,6), ylab=NULL)
> print(latticeplot)
```

3.3 GGPLOT2 Graphics

Use of the GGPLOT2 package requires that the package be loaded. Entering

```
> library(ggplot2)
```

accomplishes this. The structure of ggplot is quite different from standard R and lattice graphics. To generate a boxplot of metabolic rate that allows a comparison by sex (Figure 14) one enters the following commands.

```
> bw = ggplot(bps5.4.9,aes(sex,rate))
> bw = bw + ylab("Metabolic Rate (calories)") + xlab("Sex")
> bw = bw + geom_boxplot() + coord_flip()
> print(bw)
```

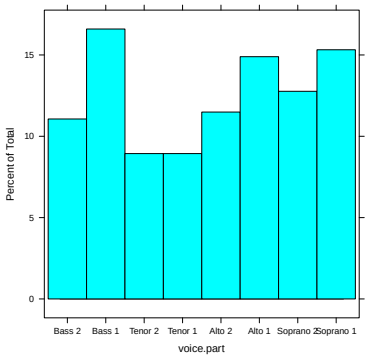

Figure 12: Bargraph generated using `histogram` on the factor variable `voice.part` from the `singer` data frame.

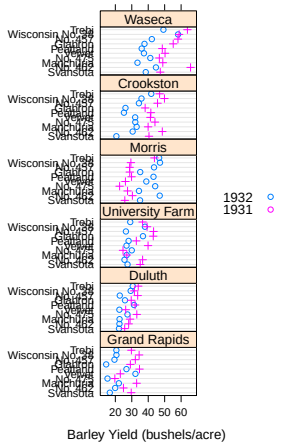

Figure 13: Density plot of the `barley` data showing the reversal of year for one of the study sites.

A histogram of metabolic rate (Figure 15) is made by entering the following code.

```
> plt = ggplot(bps5.4.9, aes(x=rate))
> plt = plt + xlab("Metabolic Rate  (calories)")
> plt = plt + geom_histogram(binwidth=200)
> print(plt)
```

The sentax for a bar chart is similar to that of a histogram. Figure 16 shows a bar chart of the `sex` variable from the BPS5e problem 4.9 data.

```
> plt = ggplot(bps5.4.9, aes(x=sex))
> plt = plt + geom_bar()
> plt = plt + xlab("Sex")
> print(plt)
```

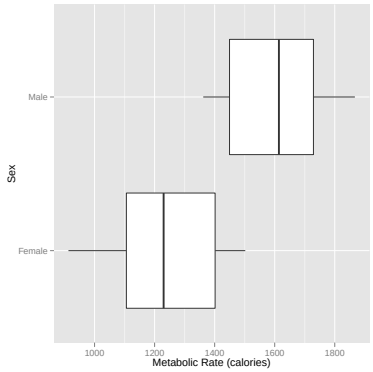

Figure 14: Boxplots of metabolic rate by sex for the data from BPS5 problem 4.9.

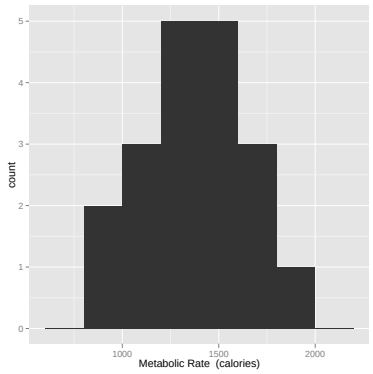

Figure 15: Histogram of metabolic rate for the data from BPS5 problem 4.9.

GGPLOT2 also provides scatterplots that can be enhanced with things like LOESS smooths (Figure 17).

```
> plt = ggplot(bps5.4.9, aes(mass, rate, shape=sex, linetype=sex))
> plt = plt + xlab("Mass (kilograms)") + ylab("Metabolic Rate (calories)")
> plt = plt + geom_point(size=3)
> plt = plt + stat_smooth(span=0.8, colour="black", lwd=0.25)
> print(plt)
```

As with the lattice package, it is possible to create separate plots for each of the sexes by using (Figure 18).

```
> plt = ggplot(bps5.4.9, aes(mass, rate)) + facet_grid(sex~.)
> plt = plt + xlab("Mass (kilograms)") + ylab("Metabolic Rate (calories)")
> plt = plt + geom_point(size=3)
> plt = plt + stat_smooth(span=0.8, colour="black", lwd=0.25)
> plt = plt + theme_bw()
> print(plt)
```

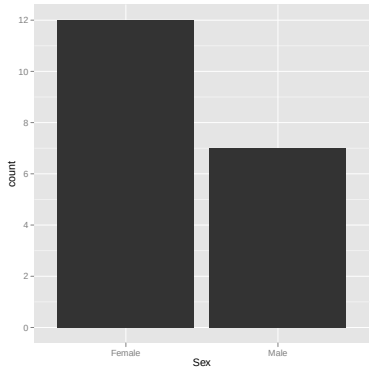

Figure 16: Bar chart of sex for the data from BPS5 problem 4.9.

4 Simple Univariate Descriptives

Summary statistics for the `htwt` data can be obtained via the `summary` function.

```
> summary(htwt)
```

```
     Height          Weight          Group
 Min.   :51.0   Min.   : 82.0   Min.   :1.00
 1st Qu.:56.0   1st Qu.:108.2   1st Qu.:1.00
 Median :59.5   Median :123.5   Median :2.00
 Mean   :62.1   Mean   :139.6   Mean   :1.55
 3rd Qu.:68.0   3rd Qu.:166.8   3rd Qu.:2.00
 Max.   :79.0   Max.   :228.0   Max.   :2.00
```

```
> summary(subset(htwt,Group=="Male"))
```

```
     Height         Weight         Group
 Min.   : NA   Min.   : NA   Min.   : NA
 1st Qu.: NA   1st Qu.: NA   1st Qu.: NA
 Median : NA   Median : NA   Median : NA
 Mean   :NaN   Mean   :NaN   Mean   :NaN
 3rd Qu.: NA   3rd Qu.: NA   3rd Qu.: NA
 Max.   : NA   Max.   : NA   Max.   : NA
```

```
> summary(subset(htwt,Group=="Female"))
```

```
     Height         Weight         Group
 Min.   : NA   Min.   : NA   Min.   : NA
 1st Qu.: NA   1st Qu.: NA   1st Qu.: NA
 Median : NA   Median : NA   Median : NA
 Mean   :NaN   Mean   :NaN   Mean   :NaN
 3rd Qu.: NA   3rd Qu.: NA   3rd Qu.: NA
 Max.   : NA   Max.   : NA   Max.   : NA
```

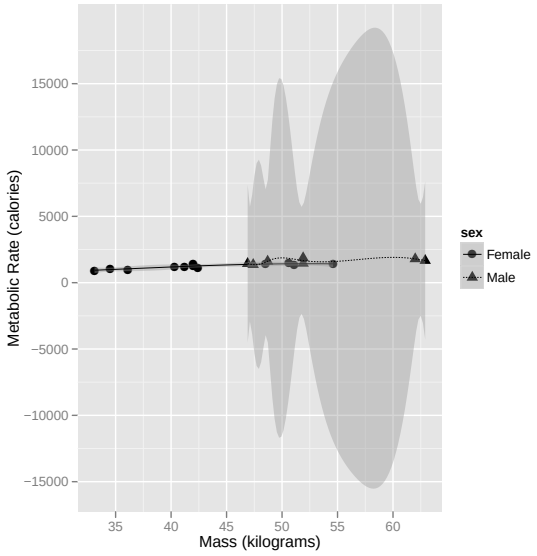

Figure 17: LOESS fits with approximate 95% confidence bounds for all data by sex.

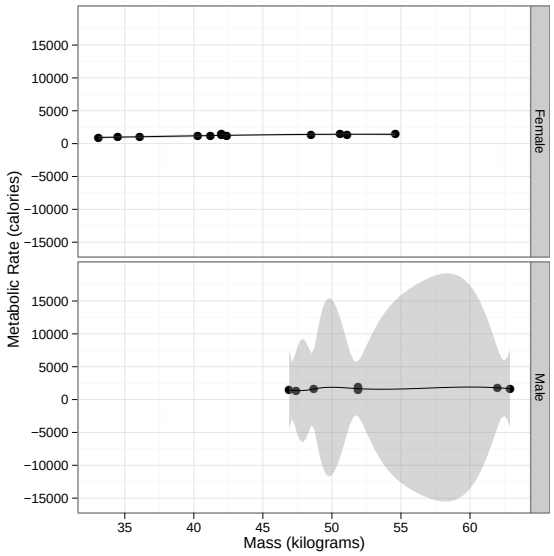

Figure 18: LOESS fits with approximate 95% confidence bounds for all data by sex.

Note that subsets of the data can be summarized using the `Group` option.

Specific values may be obtained by using specialized functions. The sample mean is computed using the `mean` function. The same value can be found through the use of the `sum` function.

```
> mean(htwt$Weight)

[1] 139.6

> sum(htwt$Weight)

[1] 2792

> length(htwt$Weight)

[1] 20

> sum(htwt$Weight)/length(htwt$Weight)

[1] 139.6

> colMeans(htwt[,1:2])

Height Weight
  62.1  139.6
```

We now compute the variance by summing the squared deviations from the mean and dividing by $n - 1$. Computing the mean once and assigning it to `xbar` and then calling `xbar` is more efficient than using `mean(htwt$Weight)` in the sum.

```
> xbar = mean(htwt$Weight)
> sum((htwt$Weight-xbar)^2)

[1] 35330.8

> sum((htwt$Weight-xbar)^2)/(length(htwt$Weight)-1)

[1] 1859.516
```

Or, we can use the `var` fucntion to compute the variance.

```
> var(htwt$Weight)

[1] 1859.516

> apply(htwt[,1:2],2,var)

    Height     Weight
  71.25263 1859.51579
```

The standard deviation is the square root of the variance. Thus, it is simple to compute the standard deviation for the `Weight` data.

```
> sqrt(var(htwt$Weight))

[1] 43.1221

> sqrt(apply(htwt[,1:2],2,var))

   Height    Weight
 8.441127 43.122103
```

When outliers or skewness are present, the above measures of centrality and spread become suspect. At these times we often turn to the median and the IQR. R makes it easy to compute these values.

We can compute the median and quartiles by sorting and then counting. The `sort` function makes this easy.

```
> sort(htwt$Weight)

 [1]  82  87  87 101 103 110 112 119 119 122 125 151 155 157 159 190 191 195 199
[20] 228
```

However, for large data sets this may be problematic. Using the R functions `median` and `quantile` are more efficient.

```
> median(htwt$Weight)

[1] 123.5

> quantile(htwt$Weight)

    0%    25%    50%    75%   100%
 82.00 108.25 123.50 166.75 228.00

> apply(htwt[,1:2],2,median)

Height Weight
  59.5  123.5
```

`Rcmdr` has the function `numSummary` which can be called from the `Rcmdr` menu –**Statistics** – **Summaries** – **Numerical Summaries**. It can also be called from the command prompt. `numSummary` computes all of the above statistics with a single call.

```
> numSummary(htwt[,"Weight"], statistics=c("mean", "sd", "quantiles"))

  mean      sd 0%    25%   50%    75% 100%  n
 139.6 43.1221 82 108.25 123.5 166.75  228 20
```

While it is possible to use mean, var, etc. and get results by group, using `numSummary` with `groups=` is easier.

```
> numSummary(htwt[,c("Height","Weight")], groups=htwt$Group,
+ statistics=c("mean", "sd", "quantiles"))

Variable: Height
      mean       sd 0% 25% 50% 75% 100%  n
1 65.00000 8.972179 52  59  64  71   79  9
2 59.72727 7.564270 51  54  58  64   76 11

Variable: Weight
  mean       sd 0%   25% 50% 75% 100%  n
1  155 48.99235 87 119.0 159 191  228  9
2  127 34.99714 82 106.5 119 153  199 11
```

5 Tables

Tables can be created both from the command line and from `Rcmdr`. We will take a look at the `hospitals` data set.

5.1 Manual Tables

The `hospitals` data frame contains three variables and 2900 observations.

```
> hospitals =
+ read.csv("http://newton.uor.edu/facultyfolder/jim_bentley/downloads/math111/hospi
> names(hospitals)

[1] "hospital"  "condition" "survival"

> hospitals[c(1:3,2900),]

     hospital condition survival
1           A      Good Survived
2           A      Good Survived
3           A      Good Survived
2900        B      Poor     Died
```

To get simple frequencies of each of the variables we can enter

```
> table(hospitals[,"hospital"])

   A    B
2100  800

> table(hospitals[,"condition"])
```

```
Good Poor
1200 1700
```

```
> table(hospitals[,"survival"])
```

```
    Died Survived
      79     2821
```

Two way tables are created by providing two columns of data. Examples might be `survival` by `hospital` or `survival` by `condition`.

```
> table(hospitals[,c("hospital","survival")])
```

```
        survival
hospital Died Survived
       A   63     2037
       B   16      784
```

```
> table(hospitals[,c("survival","condition")])
```

```
          condition
survival   Good Poor
  Died       14   65
  Survived 1186 1635
```

Notice that the order of the columns determines the rows and columns respectively.

The `table` function will also generate three-way tables.

```
> table(hospitals[,c("survival","hospital","condition")])
```

```
, , condition = Good

          hospital
survival     A   B
  Died       6   8
  Survived 594 592

, , condition = Poor

          hospital
survival     A   B
  Died      57   8
  Survived 1443 192
```

The `table` function assumes that the columns are entered as rows, columns, and tables respectively.

While the `table` function is good for getting counts, it does not generate row, column, or table percentages. `Rcmdr` does this through the use of the `xtabs`, `colPercents`, and `rowPercents` functions which are accessible through its menus—**Statistics – Contingency Tables**. These functions can also be called from the command line.

We first generate a counts table using `xtabs`.

```
> .Table = xtabs(~survival+hospital+condition, data=hospitals)
> .Table

, , condition = Good

          hospital
survival      A    B
  Died        6    8
  Survived  594  592

, , condition = Poor

          hospital
survival      A    B
  Died       57    8
  Survived 1443  192
```

To get column percents we us `colPercents` on the saved table.

```
> colPercents(.Table)

, , condition = Good

          hospital
survival     A     B
  Died       1   1.3
  Survived  99  98.7
  Total    100 100.0
  Count    600 600.0

, , condition = Poor

          hospital
survival        A   B
  Died        3.8   4
  Survived   96.2  96
  Total     100.0 100
  Count    1500.0 200
```

Row percents can be generated in a similar manner by using `rowPercents`.

```
> rowPercents(.Table)

, , condition = Good

          hospital
```

```
survival      A    B Total Count
  Died     42.9 57.1   100    14
  Survived 50.1 49.9   100  1186

, , condition = Poor

          hospital
survival      A    B Total Count
  Died     87.7 12.3   100    65
  Survived 88.3 11.7   100  1635
```

Finally, we can compute table percentages for two-way tables by using `totPercents`. We clean up by removing the table with `rm`.

```
> .Table = xtabs(~survival+hospital, data=hospitals)
> totPercents(.Table)

            A    B Total
Died      2.2  0.6   2.7
Survived 70.2 27.0  97.3
Total    72.4 27.6 100.0

> rm(.Table)
```

6 Testing The Population Mean

6.1 The One Sample Test

A simple test for the population mean of the `Weight` variable in the `htwt` data can be obtained via the `t.test` function. To compute the one sample t-test of $H_0 : \mu = 145$ we enter:

```
> t.test(htwt$Weight, mu=145, alternative='two.sided',
+        conf.level=.95)

        One Sample t-test

data:  htwt$Weight
t = -0.56, df = 19, p-value = 0.582
alternative hypothesis: true mean is not equal to 145
95 percent confidence interval:
 119.4182 159.7818
sample estimates:
mean of x
    139.6
```

An equivalent test of $H_0 : \mu = 145$ may be carried out using a linear model via the `lm` function.

```
> summary(lm((Weight-145)~1, data=htwt))

Call:
lm(formula = (Weight - 145) ~ 1, data = htwt)

Residuals:
   Min     1Q Median     3Q    Max
-57.60 -31.35 -16.10  27.15  88.40

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   -5.400      9.642   -0.56    0.582

Residual standard error: 43.12 on 19 degrees of freedom
```

Notice that adding the coefficient from the model to the hypothesized mean gives the sample mean. That is $145 + (-5.4) = 139.6$. Note, too that the p-values computed by `t.test` and `lm` are the same ($p = 0.582$).

6.2 The Two Sample Test

A simple test to compare the male and female population means of the`Weight` variable in the `htwt` data can also be obtained via the `t.test` function. To compute the two sample t-test of $H_0 : \mu_m = \mu_f$ we enter:

```
> t.test(Weight~Group, alternative='two.sided', conf.level=.95,
+         var.equal=TRUE, data=htwt)

        Two Sample t-test

data:  Weight by Group
t = 1.4903, df = 18, p-value = 0.1534
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -11.4713  67.4713
sample estimates:
mean in group 1 mean in group 2
            155             127
```

An equivalent test of $H_0 : \beta_1 = 0 = \mu_m - \mu_f$ may be carried out using a linear model via the `lm` function.

```
> summary(lm(Weight~Group, data=htwt))
```

```
Call:
lm(formula = Weight ~ Group, data = htwt)

Residuals:
   Min     1Q Median     3Q    Max
-68.00 -31.50  -6.50  31.25  73.00

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   183.00      30.58   5.984 1.17e-05 ***
Group         -28.00      18.79  -1.490    0.153
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 41.8 on 18 degrees of freedom
Multiple R-squared: 0.1098,        Adjusted R-squared: 0.06039
F-statistic: 2.221 on 1 and 18 DF,  p-value: 0.1534
```

Notice that intercept term (155) is the sample mean for the males. The sample mean for the females is the model evaluated for a female ($155+(-28) = 127$). As in the one sample problem the p-values computed by `t.test` and `lm` are the same ($p = 0.153$).

6.3 Correcting for Height

It is fairly clear from graphing `Weight` as a function of `Height` that when modeling a person's weight we should correct for height. While this cannot be accomplished using a t-test, a linear model makes the correction fairly easy.

To test for $H_0 : \beta_1 = 0$ when controlling for `Height` using the model

$$\text{Weight} = \beta_0 + \beta_1 \text{Female} + \beta_2 \text{Height} + \epsilon$$

we compute

```
> summary(lm(Weight~1+Group+Height, data=htwt))

Call:
lm(formula = Weight ~ 1 + Group + Height, data = htwt)

Residuals:
   Min     1Q Median     3Q    Max
-9.539 -6.022 -1.253  4.032 14.720

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -169.1200    15.7227 -10.756 5.26e-09 ***
Group         -1.5796     3.4779  -0.454    0.655
```

```
Height         5.0108     0.2103  23.826 1.68e-14 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.334 on 17 degrees of freedom
Multiple R-squared: 0.9741,        Adjusted R-squared: 0.9711
F-statistic: 319.9 on 2 and 17 DF,  p-value: 3.239e-14
```

Notice that as before there does note appear to be a difference between females and males ($p = 0.655$). However, it is clear that `Height` is predictive of `Weight` ($p < 0.001$).

6.4 Interaction Terms

At this point we may be convinced that no differences exist in the weights of our two groups. Clearly the means for this sample are not significantly different. A little more insight may be gained by including an interaction term.

We now fit the model

$$\text{Weight} = \beta_0 + \beta_1 \text{Female} + \beta_2 \text{Height} + \beta_3 \text{Female*Height} + \epsilon$$

```
> lm.htwt = lm(Weight~1+Group*Height, data=htwt)
> summary(lm.htwt)

Call:
lm(formula = Weight ~ 1 + Group * Height, data = htwt)

Residuals:
   Min     1Q Median     3Q    Max
-9.968 -3.413 -1.104  2.697 13.163

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  -252.7467    37.1333  -6.806 4.22e-06 ***
Group          54.4858    23.2997   2.338   0.0327 *
Height          6.3360     0.5766  10.989 7.28e-09 ***
Group:Height   -0.9013     0.3713  -2.427   0.0274 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.463 on 16 degrees of freedom
Multiple R-squared: 0.9811,        Adjusted R-squared: 0.9775
F-statistic: 276.6 on 3 and 16 DF,  p-value: 5.425e-14
```

It is now clear that not only is height predictive of weight ($p < 0.0001$), more importantly, females and males put weight on differently. Since the interaction term is

significant ($p = 0.0274$) this indicates that their slopes are different with the women putting on about one pound less per inch than the men.

Diagnostic plots can be generated by using the `plot` function on the `lm` object, `lm.htwt`. Figure 19 shows the four diagnostic plots that are the default. The analysis of variance table may also be generated.

```
> # Set up the page to take all four images
> par(mfrow=c(2,2))
> plot(lm.htwt)
> anova(lm.htwt)

Analysis of Variance Table

Response: Weight
             Df  Sum Sq Mean Sq  F value    Pr(>F)
Group         1  3880.8  3880.8  92.9116 4.570e-08 ***
Height        1 30535.6 30535.6 731.0636 8.778e-15 ***
Group:Height  1   246.1   246.1   5.8921   0.02738 *
Residuals    16   668.3    41.8
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The question of mean differences is thus shown to be the wrong question. The investigator should have been looking to see if men and women put on an equivalent number of pounds for each inch difference in height. This is something that is not apparent when looking at t-tests.

7 Fitting Logistic Models Using GLM

The examples that follow are based upon data from the Titanic disaster. Importing of the data into R can be carried out using the following code.

```
> titanic =
+   read.csv("http://newton.uor.edu/facultyfolder/jim_bentley/downloads/math111/tita
> titanic$AGE=factor(titanic$AGE,labels=c('Child','Adult'))
> titanic$CLASS=factor(titanic$CLASS,labels=c('0','1','2','3'))
> titanic$SEX=factor(titanic$SEX, labels=c('Female','Male'))
> titanic$SURVIVED=factor(titanic$SURVIVED,labels=c('No','Yes'))
```

The models fitted here give results that are equivalent to those obtained by using SAS or NCSS.

7.1 CLASS

A model to test for the difference in odds of survival as determined by class may be fitted using the `glm` function with `binomial` error and `logit` link.

```
> titanic.logistic.class=glm(SURVIVED~CLASS,
+ family=binomial(logit),data=titanic)
> summary(titanic.logistic.class)

Call:
glm(formula = SURVIVED ~ CLASS, family = binomial(logit), data = titanic)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.3999  -0.7623  -0.7401   0.9702   1.6906

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.15516    0.07876 -14.667  < 2e-16 ***
CLASS1       1.66434    0.13902  11.972  < 2e-16 ***
CLASS2       0.80785    0.14375   5.620 1.91e-08 ***
CLASS3       0.06785    0.11711   0.579    0.562
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2769.5  on 2200  degrees of freedom
Residual deviance: 2588.6  on 2197  degrees of freedom
AIC: 2596.6

Number of Fisher Scoring iterations: 4
```

Note that the (log) odds of survival do not differ for classes 0 (viewed as baseline) and 3. However, classes 1 and 2 differ from 0 (and thus 3) as well as from each other. This can most easily be seen using the odds ratios.

```
> coefs=summary(titanic.logistic.class)$coef
> est=exp(coefs[,1])
> upper.ci=exp(coefs[,1]+1.96*coefs[,2])
> lower.ci<-exp(coefs[,1]-1.96*coefs[,2])
> cbind(est,lower.ci,upper.ci)

                 est  lower.ci  upper.ci
(Intercept) 0.3150074 0.2699482 0.3675878
CLASS1      5.2822069 4.0223687 6.9366366
CLASS2      2.2430799 1.6923031 2.9731124
CLASS3      1.0702008 0.8507054 1.3463295

> rm(coefs)
```

While the odds for class 3 relative to class 0 are essentially 1:1, class 1 has a 5.28:1 odds of survival and class 2 has a 2.24:1 odds of survival relative to class 0.

7.2 AGE and SEX

A model to test for the difference in odds of survival as determined by age and sex may be fitted using the `glm` function with `binomial` error and `logit` link.

```
> titanic.logistic.agesex=glm(SURVIVED~AGE*SEX,
+ family=binomial(logit),data=titanic)
> summary(titanic.logistic.agesex)

Call:
glm(formula = SURVIVED ~ AGE * SEX, family = binomial(logit),
    data = titanic)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6497  -0.6732  -0.6732   0.7699   1.7865

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)        0.4990     0.3075   1.623   0.1046
AGEAdult           0.5654     0.3269   1.729   0.0837 .
SEXMale           -0.6870     0.3970  -1.731   0.0835 .
AGEAdult:SEXMale  -1.7465     0.4167  -4.191 2.77e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2769.5  on 2200  degrees of freedom
Residual deviance: 2312.8  on 2197  degrees of freedom
AIC: 2320.8

Number of Fisher Scoring iterations: 4
```

This model may also be expressed as

```
> titanic.logistic.agesex2=glm(SURVIVED~AGE+SEX+AGE:SEX,
+ family=binomial(logit),data=titanic)
> summary(titanic.logistic.agesex2)

Call:
glm(formula = SURVIVED ~ AGE + SEX + AGE:SEX, family = binomial(logit),
    data = titanic)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6497  -0.6732  -0.6732   0.7699   1.7865
```

```
Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)        0.4990     0.3075   1.623   0.1046
AGEAdult           0.5654     0.3269   1.729   0.0837 .
SEXMale           -0.6870     0.3970  -1.731   0.0835 .
AGEAdult:SEXMale  -1.7465     0.4167  -4.191 2.77e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2769.5  on 2200  degrees of freedom
Residual deviance: 2312.8  on 2197  degrees of freedom
AIC: 2320.8

Number of Fisher Scoring iterations: 4
```

The odds associated with the model are

```
> coefs=summary(titanic.logistic.agesex2)$coef
> est=exp(coefs[,1])
> upper.ci=exp(coefs[,1]+1.96*coefs[,2])
> lower.ci<-exp(coefs[,1]-1.96*coefs[,2])
> cbind(est,lower.ci,upper.ci)

                      est   lower.ci  upper.ci
(Intercept)      1.6470588 0.90154072 3.0090740
AGEAdult         1.7601573 0.92740960 3.3406529
SEXMale          0.5030612 0.23104993 1.0953069
AGEAdult:SEXMale 0.1743855 0.07705575 0.3946531

> rm(coefs)
```

7.3 CLASS, AGE and SEX

A model to test for the difference in odds of survival as determined by class, age and sex may be fitted using the `glm` function with `binomial` error and `logit` link.

```
> titanic.logistic.classagesex=glm(SURVIVED~AGE*SEX+CLASS*SEX+CLASS:AGE,
+ family=binomial(logit),data=titanic)
> summary(titanic.logistic.classagesex)

Call:
glm(formula = SURVIVED ~ AGE * SEX + CLASS * SEX + CLASS:AGE,
    family = binomial(logit), data = titanic)
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.6771  -0.7099  -0.5952   0.2374   2.2293

Coefficients: (1 not defined because of singularities)
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)        1.86087    0.73347   2.537  0.01118 *
AGEAdult           0.03625    0.39325   0.092  0.92655
SEXMale           -2.46011    0.81614  -3.014  0.00258 **
CLASS1            17.99982  920.38680   0.020  0.98440
CLASS2            17.11036  405.66287   0.042  0.96636
CLASS3            -2.05502    0.63854  -3.218  0.00129 **
AGEAdult:SEXMale  -0.68679    0.52541  -1.307  0.19116
SEXMale:CLASS1    -1.13608    0.82048  -1.385  0.16616
SEXMale:CLASS2    -1.06807    0.74658  -1.431  0.15254
SEXMale:CLASS3     1.66387    0.65601   2.536  0.01120 *
AGEAdult:CLASS1  -16.34159  920.38645  -0.018  0.98583
AGEAdult:CLASS2  -17.19040  405.66230  -0.042  0.96620
AGEAdult:CLASS3         NA         NA      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2769.5  on 2200  degrees of freedom
Residual deviance: 2097.5  on 2189  degrees of freedom
AIC: 2121.5

Number of Fisher Scoring iterations: 15
```

The odds associated with the model are

```
> coefs=summary(titanic.logistic.classagesex)$coef
> est=exp(coefs[,1])
> upper.ci=exp(coefs[,1]+1.96*coefs[,2])
> lower.ci<-exp(coefs[,1]-1.96*coefs[,2])
> cbind(est,lower.ci,upper.ci)

                        est   lower.ci   upper.ci
(Intercept)      6.429309e+00 1.52693798 27.0711771
AGEAdult         1.036918e+00 0.47973665  2.2412280
SEXMale          8.542543e-02 0.01725325  0.4229640
CLASS1           6.564808e+07 0.00000000        Inf
CLASS2           2.697330e+07 0.00000000        Inf
CLASS3           1.280899e-01 0.03664230  0.4477617
```

```
AGEAdult:SEXMale 5.031883e-01 0.17967631  1.4091922
SEXMale:CLASS1   3.210755e-01 0.06429782  1.6033118
SEXMale:CLASS2   3.436711e-01 0.07955044  1.4847160
SEXMale:CLASS3   5.279697e+00 1.45950127 19.0991297
AGEAdult:CLASS1  7.997187e-08 0.00000000        Inf
AGEAdult:CLASS2  3.422187e-08 0.00000000        Inf
```

```
> rm(coefs)
```

8 Fitting Logistic Models Using LRM

The models fitted here are the equivalent of those fitted above using GLM. Maybe the greatest advantage of the use of LRM is the ability to generate nomograms — if you like them that is.

8.1 CLASS

A model to test for the difference in odds of survival as determined by class may be fitted using the `lrm` function.

```
> library(rms)
> dd = datadist(titanic)
> options(datadist="dd")
> attach(titanic)
> titanic.lrm.class=lrm(SURVIVED~CLASS, x=TRUE, y=TRUE)
> titanic.lrm.class
```

```
Logistic Regression Model

lrm(formula = SURVIVED ~ CLASS, x = TRUE, y = TRUE)

                      Model Likelihood     Discrimination    Rank Discrim.
                         Ratio Test           Indexes          Indexes
Obs          2201    LR chi2     180.90    R2       0.110    C       0.642
 No          1490    d.f.             3    g        0.545    Dxy     0.283
 Yes          711    Pr(> chi2) <0.0001    gr       1.725    gamma   0.386
max |deriv| 6e-12                          gp       0.124    tau-a   0.124
                                           Brier    0.200

          Coef    S.E.   Wald Z Pr(>|Z|)
Intercept -1.1552 0.0788 -14.67 <0.0001
CLASS=1    1.6643 0.1390  11.97 <0.0001
CLASS=2    0.8078 0.1438   5.62 <0.0001
CLASS=3    0.0678 0.1171   0.58 0.5624
```

```
> anova(titanic.lrm.class)
```

```
                Wald Statistics          Response: SURVIVED

 Factor     Chi-Square d.f. P
 CLASS      173.23     3    <.0001
 TOTAL      173.23     3    <.0001
```

Note that the (log) odds of survival do not differ for classes 0 (viewed as baseline) and 3. However, classes 1 and 2 differ from 0 (and thus 3) as well as from eachother. This can most easily be seen using the odds ratios.

```
> summary(titanic.lrm.class,CLASS='0')

             Effects              Response : SURVIVED

 Factor      Low High Diff. Effect S.E. Lower 0.95 Upper 0.95
 CLASS - 1:0 1   2    NA    1.66   0.14  1.39       1.94
  Odds Ratio 1   2    NA    5.28     NA  4.02       6.94
 CLASS - 2:0 1   3    NA    0.81   0.14  0.53       1.09
  Odds Ratio 1   3    NA    2.24     NA  1.69       2.97
 CLASS - 3:0 1   4    NA    0.07   0.12 -0.16       0.30
  Odds Ratio 1   4    NA    1.07     NA  0.85       1.35

> summary(titanic.lrm.class,CLASS='3')

             Effects              Response : SURVIVED

 Factor      Low High Diff. Effect S.E. Lower 0.95 Upper 0.95
 CLASS - 0:3 4   1    NA    -0.07  0.12 -0.30       0.16
  Odds Ratio 4   1    NA     0.93    NA  0.74       1.18
 CLASS - 1:3 4   2    NA     1.60  0.14  1.31       1.88
  Odds Ratio 4   2    NA     4.94    NA  3.72       6.54
 CLASS - 2:3 4   3    NA     0.74  0.15  0.45       1.03
  Odds Ratio 4   3    NA     2.10    NA  1.57       2.80

> summary(titanic.lrm.class,CLASS='1')

             Effects              Response : SURVIVED

 Factor      Low High Diff. Effect S.E. Lower 0.95 Upper 0.95
 CLASS - 0:1 2   1    NA    -1.66  0.14 -1.94      -1.39
  Odds Ratio 2   1    NA     0.19    NA  0.14       0.25
 CLASS - 2:1 2   3    NA    -0.86  0.17 -1.18      -0.53
  Odds Ratio 2   3    NA     0.42    NA  0.31       0.59
 CLASS - 3:1 2   4    NA    -1.60  0.14 -1.88      -1.31
  Odds Ratio 2   4    NA     0.20    NA  0.15       0.27

> summary(titanic.lrm.class,CLASS='2')
```

```
             Effects              Response : SURVIVED

 Factor      Low High Diff. Effect S.E. Lower 0.95 Upper 0.95
 CLASS - 0:2 3   1    NA    -0.81  0.14 -1.09      -0.53
  Odds Ratio 3   1    NA     0.45    NA  0.34       0.59
 CLASS - 1:2 3   2    NA     0.86  0.17  0.53       1.18
  Odds Ratio 3   2    NA     2.35    NA  1.70       3.26
 CLASS - 3:2 3   4    NA    -0.74  0.15 -1.03      -0.45
  Odds Ratio 3   4    NA     0.48    NA  0.36       0.64
```

While the odds for class 3 relative to class 0 are essentially 1:1, class 1 has a 5.28:1 odds of survival and class 2 has a 2.24:1 odds of survival relative to class 0.

The probability of survival for the different classes may be plotted (Figure 20).

```
> print(plot(Predict(titanic.lrm.class, fun=plogis),
+            ylab='Probability of Survival'))
```

And we should validate the model.

```
> validate(titanic.lrm.class, B=80)
```

```
          index.orig training   test optimism index.corrected  n
Dxy           0.2833   0.2833 0.2798   0.0034          0.2799 80
R2            0.1102   0.1089 0.1090  -0.0001          0.1103 80
Intercept     0.0000   0.0000 0.0134  -0.0134          0.0134 80
Slope         1.0000   1.0000 1.0088  -0.0088          1.0088 80
Emax          0.0000   0.0000 0.0044   0.0044          0.0044 80
D             0.0817   0.0808 0.0808   0.0000          0.0818 80
U            -0.0009  -0.0009 0.0002  -0.0011          0.0002 80
Q             0.0826   0.0817 0.0806   0.0011          0.0816 80
B             0.1998   0.1995 0.2002  -0.0007          0.2004 80
g             0.5450   0.5463 0.5469  -0.0006          0.5455 80
gp            0.1240   0.1236 0.1241  -0.0004          0.1244 80
```

A nomogram may be helpful at this point (Figure 21.

```
> nom = nomogram(titanic.lrm.class, fun=plogis)
> print(plot(nom))

NULL
```

8.2 AGE and SEX

A model to test for the difference in odds of survival as determined by age and sex may be fitted using the `lmr` function.

```
> dd = datadist(titanic)
> options(datadist="dd")
> attach(titanic)

The following object(s) are masked from 'titanic (position 4)':

    AGE, CLASS, SEX, SURVIVED

> titanic.lrm.agesex=lrm(SURVIVED~AGE*SEX, x=TRUE, y=TRUE)
> titanic.lrm.agesex

Logistic Regression Model

lrm(formula = SURVIVED ~ AGE * SEX, x = TRUE, y = TRUE)

                      Model Likelihood     Discrimination    Rank Discrim.
                         Ratio Test           Indexes          Indexes
Obs          2201    LR chi2     456.68    R2       0.262    C       0.713
 No          1490    d.f.             3    g        0.841    Dxy     0.427
 Yes          711    Pr(> chi2) <0.0001    gr       2.320    gamma   0.787
max |deriv| 1e-10                          gp       0.187    tau-a   0.187
                                           Brier    0.171

                     Coef    S.E.   Wald Z Pr(>|Z|)
Intercept             0.4990 0.3075  1.62  0.1046
AGE=Adult             0.5654 0.3269  1.73  0.0837
SEX=Male             -0.6870 0.3970 -1.73  0.0835
AGE=Adult * SEX=Male -1.7465 0.4167 -4.19  <0.0001

> anova(titanic.lrm.agesex)

                Wald Statistics          Response: SURVIVED

 Factor                                     Chi-Square d.f. P
 AGE  (Factor+Higher Order Factors)          23.88     2    <.0001
  All Interactions                           17.57     1    <.0001
 SEX  (Factor+Higher Order Factors)         371.97     2    <.0001
  All Interactions                           17.57     1    <.0001
 AGE * SEX  (Factor+Higher Order Factors)    17.57     1    <.0001
 TOTAL                                      391.59     3    <.0001
```

The odds associated with the model are

```
> summary(titanic.lrm.agesex, AGE='Adult', SEX='Male')
```

```
             Effects              Response : SURVIVED

 Factor            Low High Diff. Effect S.E. Lower 0.95 Upper 0.95
 AGE - Child:Adult 2   1    NA     1.18  0.26 0.67        1.69
  Odds Ratio       2   1    NA     3.26    NA 1.96        5.41
 SEX - Female:Male 2   1    NA     2.43  0.13 2.19        2.68
  Odds Ratio       2   1    NA    11.40    NA 8.89       14.61

Adjusted to: AGE=Adult SEX=Male
```

```
> summary(titanic.lrm.agesex, AGE='Child', SEX='Female')
```

```
             Effects              Response : SURVIVED

 Factor            Low High Diff. Effect S.E. Lower 0.95 Upper 0.95
 AGE - Adult:Child 1   2    NA     0.57  0.33 -0.08      1.21
  Odds Ratio       1   2    NA     1.76    NA  0.93      3.34
 SEX - Male:Female 1   2    NA    -0.69  0.40 -1.47      0.09
  Odds Ratio       1   2    NA     0.50    NA  0.23      1.10

Adjusted to: AGE=Child SEX=Female
```

The probability of survival for the different combinations of sex and age group may be plotted (Figure 22).

```
> Predict(titanic.lrm.agesex, fun=plogis,
+         AGE=c('Child','Adult'), SEX=c('Female','Male'))

    AGE    SEX      yhat     lower     upper
1 Child Female 0.6222222 0.4741134 0.7505638
2 Adult Female 0.7435294 0.6998704 0.7828084
3 Child   Male 0.4531250 0.3362143 0.5754460
4 Adult   Male 0.2027594 0.1841419 0.2227454

Response variable (y):

Limits are 0.95 confidence limits
```

```
> print(plot(Predict(titanic.lrm.agesex, fun=plogis,
+              AGE=c('Child','Adult'), SEX=c('Female','Male'))))
```

And we should validate the model.

```
> validate(titanic.lrm.agesex, B=80)
```

```
          index.orig training    test optimism index.corrected  n
Dxy           0.4267   0.4277  0.4264   0.0013          0.4254 80
R2            0.2618   0.2636  0.2608   0.0029          0.2589 80
Intercept     0.0000   0.0000 -0.0012   0.0012         -0.0012 80
Slope         1.0000   1.0000  0.9951   0.0049          0.9951 80
Emax          0.0000   0.0000  0.0013   0.0013          0.0013 80
D             0.2070   0.2088  0.2062   0.0026          0.2044 80
U            -0.0009  -0.0009 -0.0001  -0.0008         -0.0001 80
Q             0.2079   0.2097  0.2062   0.0035          0.2045 80
B             0.1713   0.1708  0.1716  -0.0008          0.1720 80
g             0.8414   0.8451  0.8370   0.0080          0.8334 80
gp            0.1867   0.1871  0.1858   0.0013          0.1854 80
```

A nomogram may be helpful at this point (Figure 23).

```
> nom = nomogram(titanic.lrm.agesex, fun=plogis)
> print(plot(nom))

NULL
```

8.3 CLASS, AGE and SEX

A model to test for the difference in odds of survival as determined by class, age and sex may be fitted using the `lmr` function.

```
> dd = datadist(titanic)
> options(datadist="dd")
> attach(titanic)

The following object(s) are masked from 'titanic (position 4)':

    AGE, CLASS, SEX, SURVIVED
The following object(s) are masked from 'titanic (position 6)':

    AGE, CLASS, SEX, SURVIVED

> titanic.lrm.classagesex=lrm(SURVIVED~CLASS*SEX+AGE*SEX, x=TRUE, y=TRUE)
> titanic.lrm.classagesex

Logistic Regression Model

lrm(formula = SURVIVED ~ CLASS * SEX + AGE * SEX, x = TRUE, y = TRUE)

                      Model Likelihood     Discrimination    Rank Discrim.
                         Ratio Test            Indexes          Indexes
Obs          2201    LR chi2     634.70    R2       0.350    C       0.766
```

```
 No          1490    d.f.             9    g        1.341    Dxy     0.532
 Yes          711    Pr(> chi2) <0.0001    gr       3.823    gamma   0.638
max |deriv| 5e-10                          gp       0.233    tau-a   0.233
                                           Brier    0.157

                    Coef    S.E.   Wald Z Pr(>|Z|)
Intercept            2.0775 0.7171  2.90  0.0038
CLASS=1              1.6642 0.8003  2.08  0.0376
CLASS=2              0.0497 0.6874  0.07  0.9424
CLASS=3             -2.0894 0.6381 -3.27  0.0011
SEX=Male            -1.7888 0.7728 -2.31  0.0206
AGE=Adult           -0.1803 0.3618 -0.50  0.6182
CLASS=1 * SEX=Male  -1.1033 0.8199 -1.35  0.1784
CLASS=2 * SEX=Male  -0.7647 0.7271 -1.05  0.2929
CLASS=3 * SEX=Male   1.5623 0.6562  2.38  0.0173
SEX=Male * AGE=Adult -1.3581 0.4551 -2.98  0.0028
```

```
> anova(titanic.lrm.classagesex)

                Wald Statistics          Response: SURVIVED

 Factor                                          Chi-Square d.f. P
 CLASS  (Factor+Higher Order Factors)             124.28     6    <.0001
  All Interactions                                 48.25     3    <.0001
 SEX  (Factor+Higher Order Factors)               254.38     5    <.0001
  All Interactions                                 63.47     4    <.0001
 AGE  (Factor+Higher Order Factors)                31.30     2    <.0001
  All Interactions                                  8.91     1    0.0028
 CLASS * SEX  (Factor+Higher Order Factors)        48.25     3    <.0001
 SEX * AGE  (Factor+Higher Order Factors)          8.91     1    0.0028
 TOTAL INTERACTION                                 63.47     4    <.0001
 TOTAL                                            311.38     9    <.0001
```

The odds associated with the model are

```
> summary(titanic.lrm.classagesex, CLASS='0', AGE='Adult', SEX='Male')

              Effects              Response : SURVIVED

 Factor            Low High Diff. Effect S.E. Lower 0.95 Upper 0.95
 CLASS - 1:0       1   2    NA     0.56  0.18  0.21       0.91
  Odds Ratio       1   2    NA     1.75    NA  1.24       2.48
 CLASS - 2:0       1   3    NA    -0.72  0.24 -1.18      -0.25
  Odds Ratio       1   3    NA     0.49    NA  0.31       0.78
 CLASS - 3:0       1   4    NA    -0.53  0.15 -0.83      -0.23
  Odds Ratio       1   4    NA     0.59    NA  0.44       0.80
```

```
 SEX - Female:Male 2   1    NA     3.15  0.62  1.92       4.37
  Odds Ratio       2   1    NA    23.26    NA  6.84      79.12
 AGE - Child:Adult 2   1    NA     1.54  0.28  1.00       2.08
  Odds Ratio       2   1    NA     4.66    NA  2.71       8.00

Adjusted to: CLASS=0 SEX=Male AGE=Adult
```

```
> summary(titanic.lrm.classagesex, CLASS='3', AGE='Child', SEX='Female')

             Effects              Response : SURVIVED

 Factor            Low High Diff. Effect S.E. Lower 0.95 Upper 0.95
 CLASS - 0:3       4   1    NA     2.09  0.64  0.84        3.34
  Odds Ratio       4   1    NA     8.08    NA  2.31       28.22
 CLASS - 1:3       4   2    NA     3.75  0.53  2.72        4.79
  Odds Ratio       4   2    NA    42.67    NA 15.11      120.56
 CLASS - 2:3       4   3    NA     2.14  0.33  1.49        2.79
  Odds Ratio       4   3    NA     8.49    NA  4.45       16.20
 SEX - Male:Female 1   2    NA    -0.23  0.42 -1.06        0.60
  Odds Ratio       1   2    NA     0.80    NA  0.35        1.83
 AGE - Adult:Child 1   2    NA    -0.18  0.36 -0.89        0.53
  Odds Ratio       1   2    NA     0.83    NA  0.41        1.70

Adjusted to: CLASS=3 SEX=Female AGE=Child
```

The probability of survival for the different combinations of sex and age group may be plotted (Figure 24).

```
> Predict(titanic.lrm.classagesex, fun=plogis,
+         CLASS=c('0','1','2','3'), AGE=c('Child','Adult'),
+         SEX=c('Female','Male'))

   CLASS   AGE    SEX      yhat      lower     upper
1      0 Child Female 0.8886935 0.66194638 0.9701987
2      1 Child Female 0.9768348 0.92575376 0.9930367
3      2 Child Female 0.8935160 0.78056016 0.9519104
4      3 Child Female 0.4970148 0.33827964 0.6563539
5      0 Adult Female 0.8695652 0.66454828 0.9573283
6      1 Adult Female 0.9723831 0.92874210 0.9895961
7      2 Adult Female 0.8750999 0.79597143 0.9263784
8      3 Adult Female 0.4520760 0.37865906 0.5276396
9      0 Child   Male 0.5716729 0.43150500 0.7012113
10     1 Child   Male 0.7004700 0.55927805 0.8116614
11     2 Child   Male 0.3949969 0.25626440 0.5529915
12     3 Child   Male 0.4406809 0.32190559 0.5666583
13     0 Adult   Male 0.2227378 0.19619893 0.2517422
```

```
14     1 Adult   Male 0.3342723 0.26910345 0.4064468
15     2 Adult   Male 0.1229466 0.08312897 0.1781310
16     3 Adult   Male 0.1446912 0.11604927 0.1789709

Response variable (y):

Limits are 0.95 confidence limits
```

```
> print(plot(Predict(titanic.lrm.classagesex, fun=plogis,
+       CLASS=c('0','1','2','3'),
+       SEX=c('Female','Male'),
+       AGE=c('Child','Adult')),
+       pch=c(2,1),col=c(1,2),layout=c(1,2)))
```

And we should validate the model.

```
> validate(titanic.lrm.classagesex, B=80)
```

```
          index.orig training    test optimism index.corrected  n
Dxy           0.5322   0.5320  0.5288   0.0031          0.5291 80
R2            0.3500   0.3528  0.3459   0.0070          0.3430 80
Intercept     0.0000   0.0000 -0.0233   0.0233         -0.0233 80
Slope         1.0000   1.0000  0.9735   0.0265          0.9735 80
Emax          0.0000   0.0000  0.0100   0.0100          0.0100 80
D             0.2879   0.2906  0.2840   0.0066          0.2814 80
U            -0.0009  -0.0009  0.0002  -0.0011          0.0002 80
Q             0.2888   0.2915  0.2838   0.0077          0.2811 80
B             0.1571   0.1563  0.1578  -0.0015          0.1585 80
g             1.3411   1.3732  1.3352   0.0381          1.3030 80
gp            0.2326   0.2330  0.2304   0.0026          0.2300 80
```

A nomogram may be helpful at this point (Figure 25).

```
> nom = nomogram(titanic.lrm.classagesex, fun=plogis)
> print(plot(nom))

NULL
```

9 Fitting CART

The CARTs fitted here are analogous to the logistic models fitted in SAS and R.

9.1 CLASS

A classification tree to look at the predictive nature of class when looking at survival may be fitted using the `rpart` function.

```
> #install.packages(c("rpart", "rpart.plot", "rpartOrdinal"))
> library(rpart)
> library(rpart.plot)
> library(rpartOrdinal)
> titanic.rpart.class=rpart(SURVIVED~CLASS,data=titanic)
> summary(titanic.rpart.class)

Call:
rpart(formula = SURVIVED ~ CLASS, data = titanic)
  n= 2201

          CP nsplit rel error    xerror       xstd
1 0.05696203      0 1.0000000 1.0000000 0.03085662
2 0.01000000      2 0.8860759 0.8860759 0.02982488

Node number 1: 2201 observations,    complexity param=0.05696203
  predicted class=No   expected loss=0.323035
    class counts:  1490   711
   probabilities: 0.677 0.323
  left son=2 (1591 obs) right son=3 (610 obs)
  Primary splits:
      CLASS splits as  LRRL, improve=69.6841, (0 missing)

Node number 2: 1591 observations
  predicted class=No   expected loss=0.2451288
    class counts:  1201   390
   probabilities: 0.755 0.245

Node number 3: 610 observations,    complexity param=0.05696203
  predicted class=Yes  expected loss=0.4737705
    class counts:   289   321
   probabilities: 0.474 0.526
  left son=6 (285 obs) right son=7 (325 obs)
  Primary splits:
      CLASS splits as  -RL-, improve=13.46678, (0 missing)

Node number 6: 285 observations
  predicted class=No   expected loss=0.4140351
    class counts:   167   118
   probabilities: 0.586 0.414
```

```
Node number 7: 325 observations
  predicted class=Yes  expected loss=0.3753846
    class counts:   122   203
   probabilities: 0.375 0.625
```

A plot of the tree (Figure 26) may be created using

```
> plot(titanic.rpart.class)
> text(titanic.rpart.class)
```

9.2 AGE and SEX

A classification tree to look at the predictive nature of age and sex when looking at survival may be fitted using the `rpart` function.

```
> titanic.rpart.agesex=rpart(SURVIVED~AGE+SEX,data=titanic)
> summary(titanic.rpart.agesex)

Call:
rpart(formula = SURVIVED ~ AGE + SEX, data = titanic)
  n= 2201

         CP nsplit rel error    xerror       xstd
1 0.3066104      0 1.0000000 1.0000000 0.03085662
2 0.0100000      1 0.6933896 0.6933896 0.02750982

Node number 1: 2201 observations,    complexity param=0.3066104
  predicted class=No   expected loss=0.323035
    class counts:  1490   711
   probabilities: 0.677 0.323
  left son=2 (1731 obs) right son=3 (470 obs)
  Primary splits:
      SEX splits as  RL, improve=199.821600, (0 missing)
      AGE splits as  RL, improve=  9.165241, (0 missing)

Node number 2: 1731 observations
  predicted class=No   expected loss=0.2120162
    class counts:  1364   367
   probabilities: 0.788 0.212

Node number 3: 470 observations
  predicted class=Yes  expected loss=0.2680851
    class counts:   126   344
   probabilities: 0.268 0.732
```

A plot of the tree (Figure 27) may be created using

```
> plot(titanic.rpart.agesex)
> text(titanic.rpart.agesex)
```

9.3 CLASS, AGE and SEX

A classification tree to look at the predictive nature of class, age and sex when looking at survival may be fitted using the `rpart` function.

```
> titanic.rpart.classagesex=rpart(SURVIVED~CLASS+AGE+SEX,data=titanic)
> summary(titanic.rpart.classagesex)

Call:
rpart(formula = SURVIVED ~ CLASS + AGE + SEX, data = titanic)
  n= 2201

          CP nsplit rel error    xerror       xstd
1 0.30661041      0 1.0000000 1.0000000 0.03085662
2 0.02250352      1 0.6933896 0.6933896 0.02750982
3 0.01125176      2 0.6708861 0.6976090 0.02756915
4 0.01000000      4 0.6483826 0.6779184 0.02728864

Node number 1: 2201 observations,    complexity param=0.3066104
  predicted class=No   expected loss=0.323035
    class counts:  1490   711
   probabilities: 0.677 0.323
  left son=2 (1731 obs) right son=3 (470 obs)
  Primary splits:
      SEX   splits as  RL,   improve=199.821600, (0 missing)
      CLASS splits as  LRRL, improve= 69.684100, (0 missing)
      AGE   splits as  RL,   improve=  9.165241, (0 missing)

Node number 2: 1731 observations,    complexity param=0.01125176
  predicted class=No   expected loss=0.2120162
    class counts:  1364   367
   probabilities: 0.788 0.212
  left son=4 (1667 obs) right son=5 (64 obs)
  Primary splits:
      AGE   splits as  RL,   improve=7.726764, (0 missing)
      CLASS splits as  LRLL, improve=7.046106, (0 missing)

Node number 3: 470 observations,    complexity param=0.02250352
  predicted class=Yes  expected loss=0.2680851
    class counts:   126   344
   probabilities: 0.268 0.732
  left son=6 (196 obs) right son=7 (274 obs)
```

```
  Primary splits:
      CLASS splits as  RRRL, improve=50.015320, (0 missing)
      AGE   splits as  LR,   improve= 1.197586, (0 missing)
  Surrogate splits:
      AGE splits as  LR, agree=0.619, adj=0.087, (0 split)

Node number 4: 1667 observations
  predicted class=No   expected loss=0.2027594
    class counts:  1329   338
   probabilities: 0.797 0.203

Node number 5: 64 observations,    complexity param=0.01125176
  predicted class=No   expected loss=0.453125
    class counts:    35    29
   probabilities: 0.547 0.453
  left son=10 (48 obs) right son=11 (16 obs)
  Primary splits:
      CLASS splits as  -RRL, improve=12.76042, (0 missing)

Node number 6: 196 observations
  predicted class=No   expected loss=0.4591837
    class counts:   106    90
   probabilities: 0.541 0.459

Node number 7: 274 observations
  predicted class=Yes  expected loss=0.0729927
    class counts:    20   254
   probabilities: 0.073 0.927

Node number 10: 48 observations
  predicted class=No   expected loss=0.2708333
    class counts:    35    13
   probabilities: 0.729 0.271

Node number 11: 16 observations
  predicted class=Yes  expected loss=0
    class counts:     0    16
   probabilities: 0.000 1.000
```

A plot of the tree (Figure 28) may be created using

```
> plot(titanic.rpart.classagesex)
> text(titanic.rpart.classagesex)
```

9.4 Additional Functions

The documentation for the function `rpart` shows how to prune classification trees. There are also a number of sites on the web that show how to interpret `rpart` output.

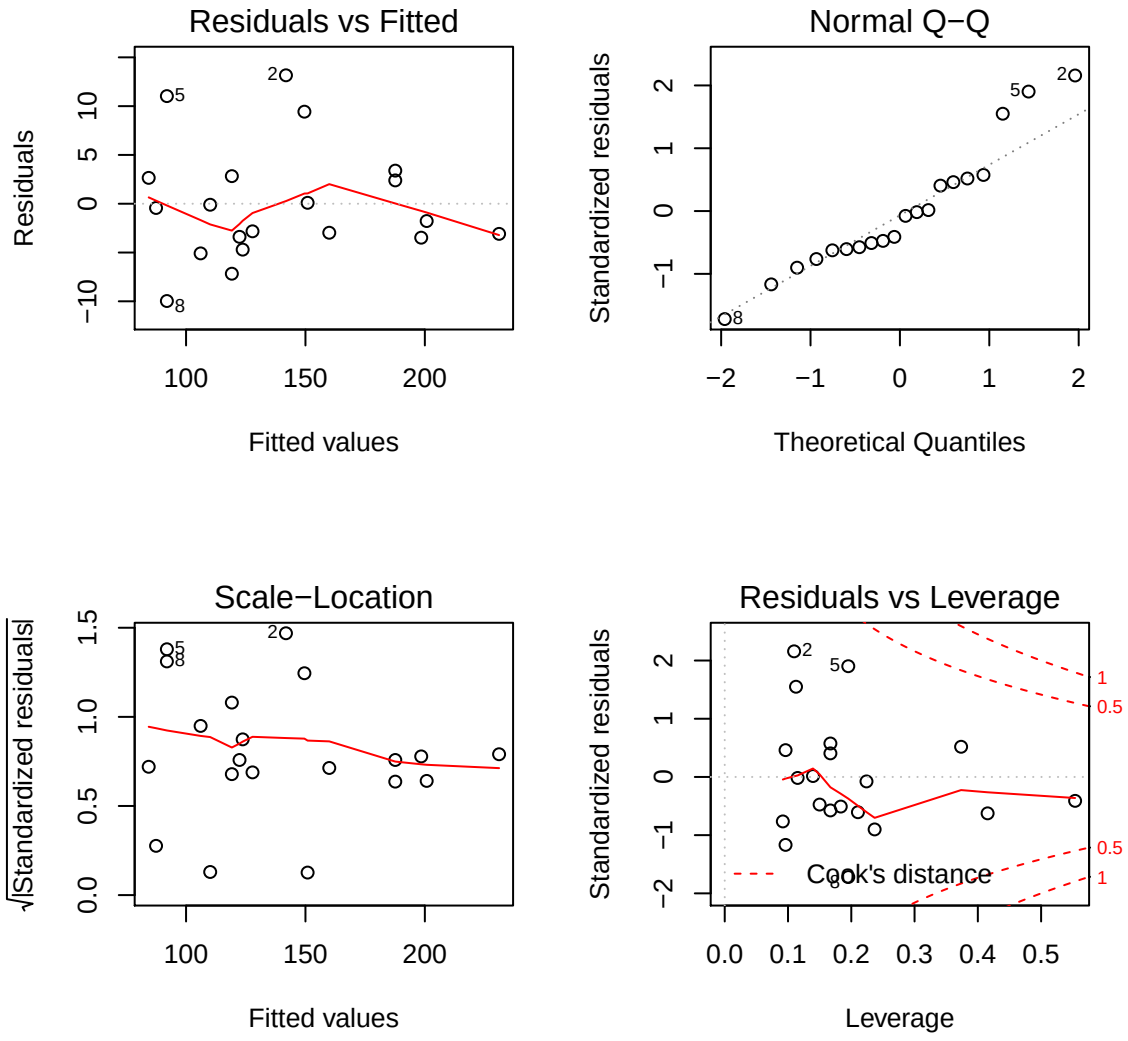

Figure 19: Default diagnostic plots for the full model fitted to the `htwt` data.

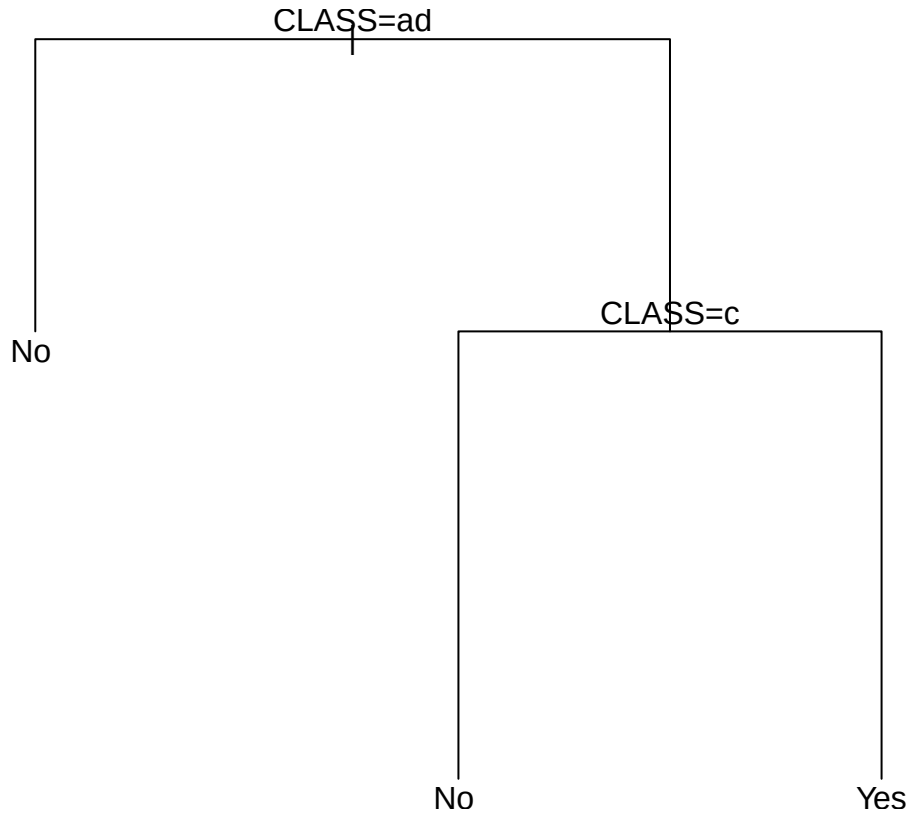

Figure 20: Estimated probability of survival (and 95% CIs) based upon class.

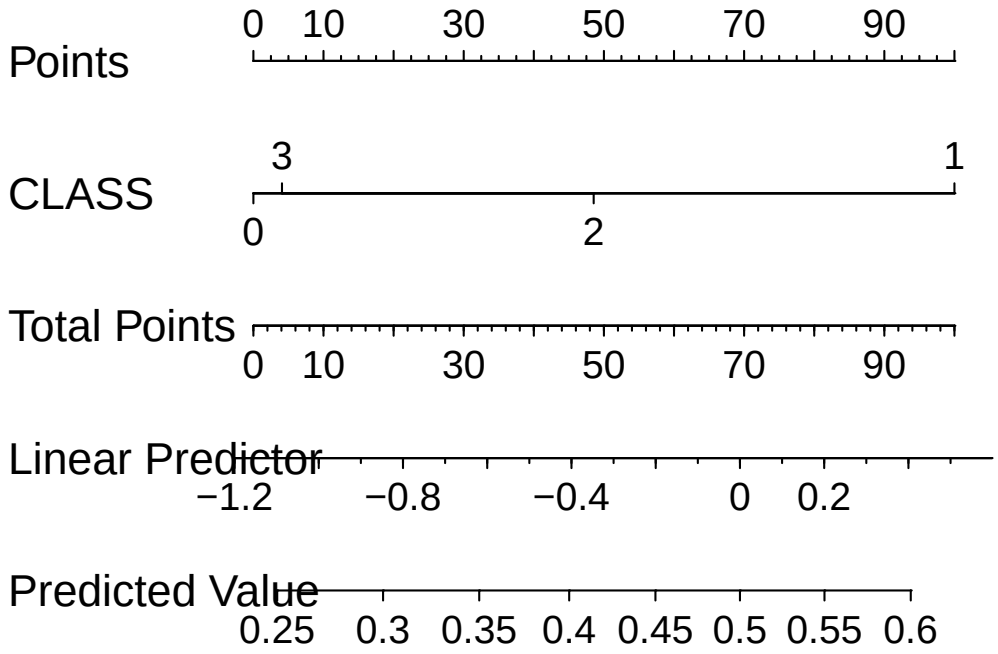

Figure 21: Estimated probability of survival based upon class.

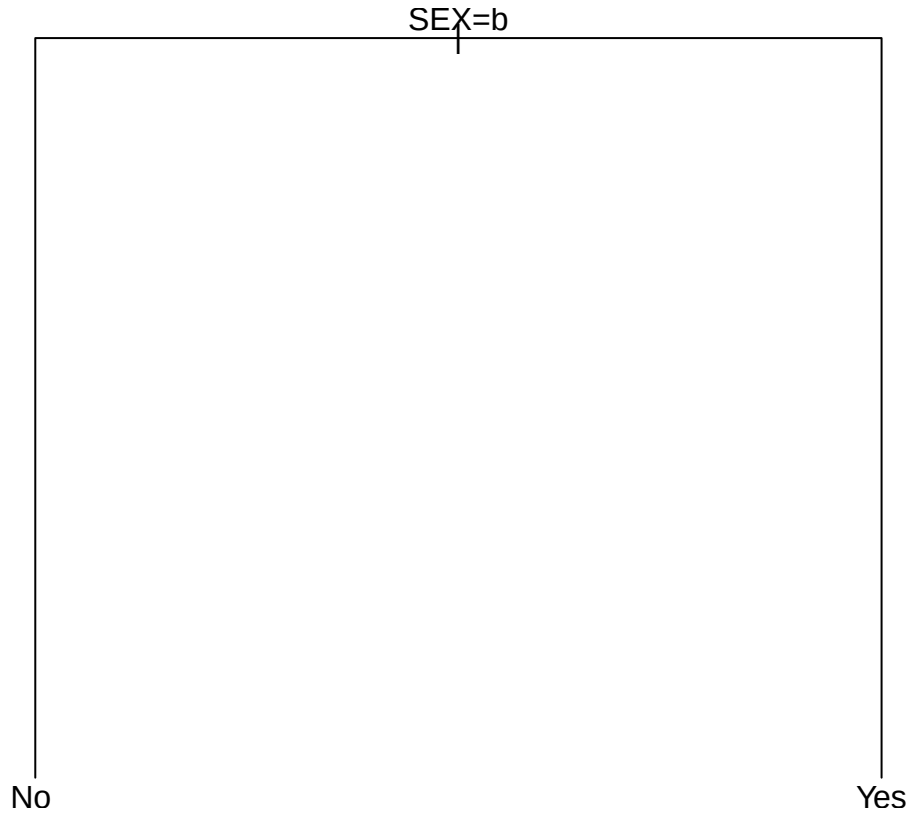

Figure 22: Estimated probability of survival based upon sex and age group.

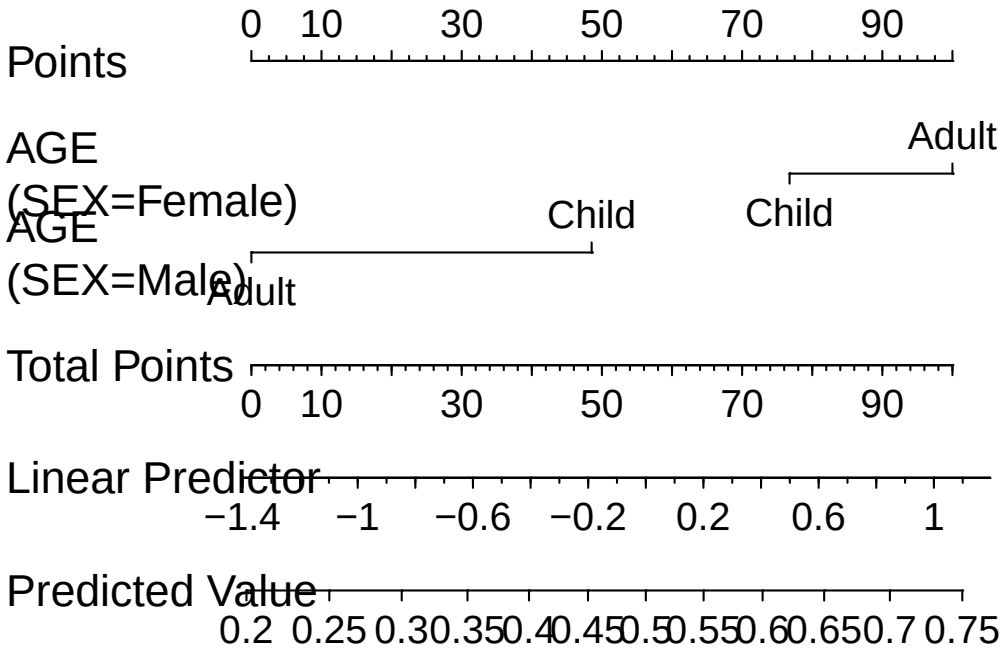

Figure 23: Nomogram for survival based upon sex and age group.

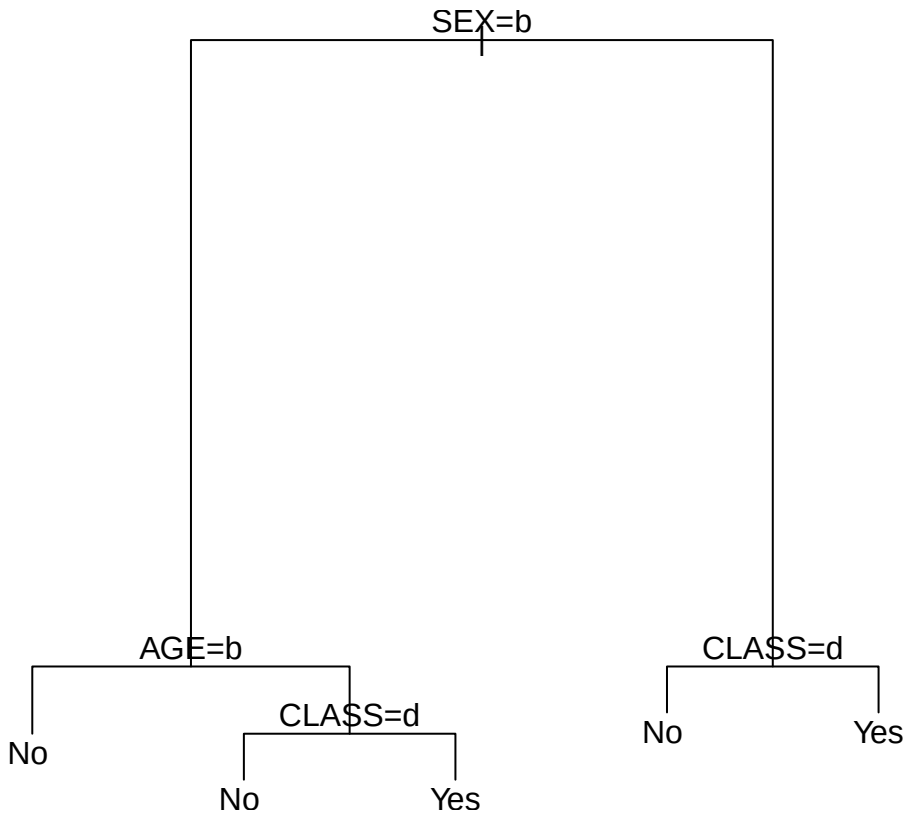

Figure 24: Estimated probability of survival based upon class, sex and age group.

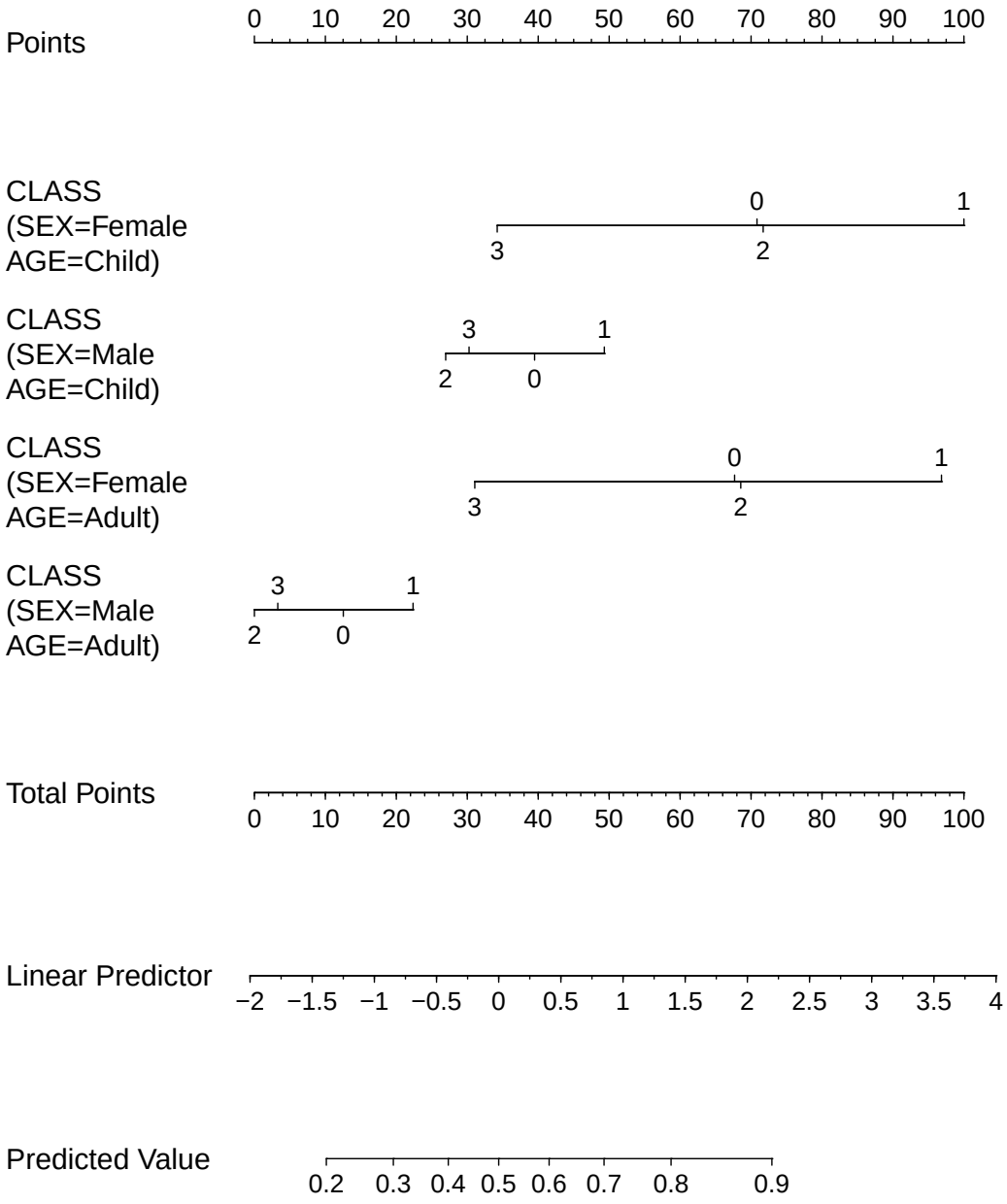

Figure 25: Nomogram for survival based upon class, sex and age group.

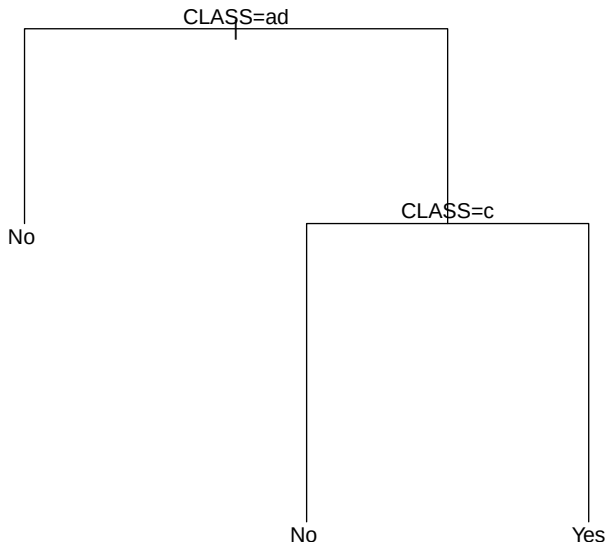

Figure 26: Classification tree for survival based upon class.

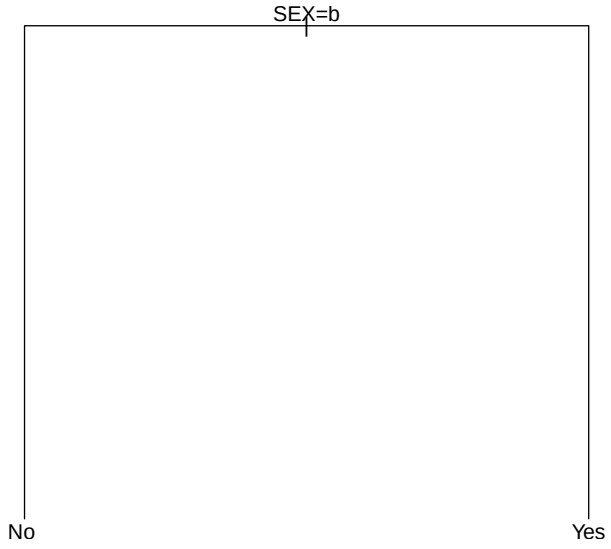

Figure 27: Classification tree for survival based upon age and sex.

Figure 28: Classification tree for survival based upon class, age and sex.