Statistics with R – Beginner Level

Section 1

Data Manipulation in R

Lesson 1 - Filtering Data Using Brackets

```
demo <- read.csv("demographics.csv")

View(demo)

### how to filter (select) your data data in base R using brackets

### a new data frame, demo2, will be created each time

######
### one filter variable
######

### select the female subjects

demo2 <- demo[demo$gender == "Female",]

View(demo2)

### retain the subjects with the income greater than 100

demo2 <- demo[demo$income > 100,]

View(demo2)
```

```
### if you want to keep only the variables 1, 3 and 7
### (age, income and gender)

demo2 <- demo[demo$income > 100, c(1,3,7)]

View(demo2)

### if you want to drop variables 6, 7 and 8
### (car category, gender and retired)

demo2 <- demo[demo$income > 100, -c(6:8)]

View(demo2)

######
### two or more filter variables
######

### select the female subjects with the income over 100

demo2 <- demo[demo$gender == "Female" & demo$income > 100,]

View(demo2)
```

Lesson 2 - Filtering Data with the Subset Command

```
demo <- read.csv("demographics.csv")

View(demo)

### how to filter (select) your data in base R using
subsets

### a new data frame, demo2, will be created each time

### keep the married subjects only
### (one filter variable)

demo2 <- subset(demo, marital == "Married")

View(demo2)
```

```
### retain the married subjects aged over 35
### (two filter variables)

demo2 <- subset(demo, marital == "Married" & age > 35)

View(demo2)

### keep the first three variables only
### (age, marital status, income)

demo2 <- subset(demo, marital == "Married" & age > 35,
select = c(1:3))

View(demo2)

### drop variables 4, 5, 6 and 8
### (education, car price, car category, retired)

demo2 <- subset(demo, marital == "Married" & age > 35,
select = -c(4:6, 8))

View(demo2)
```

Lesson 3 - Filtering Data with Dplyr

```
demo <- read.csv("demographics.csv")

View(demo)

### how to filter (select) your data using the dplyr
package

### a new data frame, demo2, will be created each time

### load the package

require(dplyr)

### keep the unmarried subjects only
### (one filter variable)

demo2 <- filter(demo, marital == "Unmarried")
```

```
View(demo2)

### keep the unmarried subjects only aged under 50
### (two filter variables)

demo2 <- filter(demo, marital == "Unmarried", age < 50)

View(demo2)

### if you want to keep some variables only,
### you must first specify the variables you want to keep

### suppose we want to keep only the first three variables
### (age, marital status, income)

demo2 <- select(demo, age, marital, income)

View(demo2)

### next we filter our new data frame demo2,
### keeping only the unmarried persons aged under 50

demo2 <- filter(demo2, marital == "Unmarried", age < 50)

View(demo2)
```

Lesson 4 - Recoding Categorical Variables

```
demo <- read.csv("demographics.csv")

View(demo)

### how to recode the categorical (factor) variables

### we want to convert the variable gender as follows
### Male = 1, Female = 2

### a new variable gender2 will be created

### first we will use the brackets (base R)

demo$gender2[demo$gender == "Male"] = "1"
demo$gender2[demo$gender == "Female"] = "2"
```

```
View (demo)

##########

### we can do the same type of recoding with the plyr
package, function revalue

### load the package

require(plyr)

### let's create a new variable, gender3

demo$gender3 = revalue(demo$gender, c("Male"="1",
"Female"="2"))

View(demo)

### important: if the variable to recode is not a factor
### we must convert it into a factor before recoding

demo$gender = factor(demo$gender)

### to recode into the same variable (without creating a
new one)
### we just use the same variable name in both sides of the
revalue function

demo$gender = revalue(demo$gender, c("Male"="1",
"Female"="2"))

View(demo)
```

Lesson 5 - Recoding Continuous Variables

```
demo <- read.csv("demographics.csv")

View(demo)

### how to recode a continous variable into a factor
### we want to create a categorical variable as follows
### subjects with income under 200 - low income
### subjects with income of 200 and more - high income

### a new variable, incat (income category), will be
created

demo$incat[demo$income<200] = "Low income"
demo$incat[demo$income>=200] = "High income"

View(demo)

### now we want to create three groups by income
### low income - under 150
### medium income - between 150 and 300
### high income - 300 and more
### so we will have two cut points: 150 and 300

### a new variable, incat2, will be created

demo$incat2 = cut(demo$income, breaks=c(-Inf, 150, 300,
Inf), labels=c("Low income", "Medium income", "High
income"))

View(demo)

### by default, the ranges are open on the left, and closed
on the right
### namely (-Inf,150], (150, 300] and (300, Inf)
### to get it conversely, use the option right=FALSE

demo$incat2 = cut(demo$income, breaks=c(-Inf, 150, 300,
Inf), labels=c("Low income", "Medium income", "High
income"), right = FALSE)
```

Lesson 6 - Sorting Data Frames

```
demo <- read.csv("demographics.csv")

View(demo)
```

```
### how to sort a data frame
### a new data frame, demo2, will be created each time
### sort by income, ascending (default)
demo2 <- demo[order(demo$income),]
View(demo2)
### sort by income, descending
demo2 <- demo[order(-demo$income),]
View(demo2)
### sort by income and age
demo2 <- demo[order(demo$income, demo$age),]
View(demo2)

### sort by income (ascending) and age (descending)
demo2 <- demo[order(demo$income, -demo$age),]
View(demo2)
```

Lesson 7 - Compute New Variables

```
math <- read.csv("math.csv")
View(math)
### how to compute a new variable
### we will create a variable that stores the difference
between the two grades

math$diff = math$grade2 - math$grade1

### another variable that stores the average of the two
grades

math$avg = (math$grade1 + math$grade2) / 2
```