Introduction to R

Session 3 – Data manipulation

Statistical Consulting Centre

19 July, 2017

1. `ifelse()` function and factor

1. Create a new variable called `pHtype` which has the value `acidity` if `pH` < 7, the value `alkalinity` if `pH` > 7 and the value `natural` if `pH` $= 7$

```
lake.df$pHtype <- with(lake.df, ifelse(pH > 7, "alkalinity", ifelse(pH < 7, "acidity", "natural")))
```

2. Convert the variable created in 1.1 into factors with appropriate levels: "acidity", "natural" and "alkalinity".

```
lake.df$pHtype <- factor(lake.df$pHtype, levels = c( "acidity", "natural", "alkalinity"))
```

3. Convert the `Calcium` in `lake.df` into factors with appropriate levels: "Low", "Medium" and "High".

```
lake.df$Calcium <- factor(lake.df$Calcium, levels = c("Low", "Medium", "High"))
```

4. Produce a two-way frequency table of `Calcium` versus `pHtype`.

```
with(lake.df, table(Calcium, pHtype))
```

```
##         pHtype
## Calcium  acidity natural alkalinity
##   Low         16       1          1
##   Medium      12       1          6
##   High         3       1         12
```

2. Join two datasets

1. `mercury.csv` contains data on mercury contamination in 53 different lakes in Florida. The mercury concentration (parts per million) in the muscle tissue of the fish sampled from that lake were taken in
   - Day1
   - Day2
   - Day3
2. Read the data into R, saving it in object named mercury.df.

```
mercury.df <- read.csv("location of your folder/Mercury.csv",
                       stringsAsFactors = FALSE)
```

3. Now combine the two datasets:`lake.df` and `mercury.df` by the `ID`, and call the new dataset `joined.df`.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
joined.df <- left_join(lake.df, mercury.df, by = "ID")
```

4. Check its dimensions using `dim()`.

```
dim(joined.df)
```

```
## [1] 53  9
```

5. Print this objects variable names to the console.

```
names(joined.df)
```

```
## [1] "ID"          "Lake"        "pH"          "Calcium"     "Chlorophyll"
## [6] "pHtype"      "Day1"        "Day2"        "Day3"
```

6. Use `str()` to check all of the variables at once.

```
str(joined.df)
```

```
## 'data.frame':    53 obs. of  9 variables:
##  $ ID         : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Lake       : chr  "Alligator" "Annie" "Apopka" "Blue Cypress" ...
##  $ pH         : num  6.1 5.1 9.1 6.9 4.6 7.3 5.4 8.1 5.8 6.4 ...
##  $ Calcium    : Factor w/ 3 levels "Low","Medium",..: 1 1 3 2 1 1 1 3 2 1 ...
##  $ Chlorophyll: num  0.7 3.2 128.3 3.5 1.8 ...
##  $ pHtype     : Factor w/ 3 levels "acidity","natural",..: 1 1 3 1 1 3 1 3 1 1 ...
##  $ Day1       : num  0.967 1.249 -0.012 0.458 1.157 ...
##  $ Day2       : num  1.2226 1.5668 0.0261 0.5409 1.2484 ...
##  $ Day3       : num  1.2236 1.6178 0.0775 0.4199 1.2179 ...
```

3. `for` loop

1. Check the one-way frequency tables of pHtype and Calcium.

```
for (i in c("pHtype", "Calcium")){
  print(i)
  print(table(joined.df[,i]))
}
```

```
## [1] "pHtype"
##
##    acidity    natural alkalinity
##         31          3         19
## [1] "Calcium"
##
##    Low Medium   High
##     18     19     16
```

2. Calculate the means and standard deviations of pH, Chlorophyll and mercury measuremnts at Day 1, 2, and 3.

```r
for (i in c("pH", "Chlorophyll", "Day1", "Day2", "Day3")){
  print(i)
  print(mean(joined.df[,i], na.rm = TRUE))
  print(sd(joined.df[,i], na.rm = TRUE))
}
```

```
## [1] "pH"
## [1] 6.590566
## [1] 1.288449
## [1] "Chlorophyll"
## [1] 23.11698
## [1] 30.81632
## [1] "Day1"
## [1] 0.5087247
## [1] 0.3257218
## [1] "Day2"
## [1] 0.523276
## [1] 0.3706001
## [1] "Day3"
## [1] 0.5341706
## [1] 0.3681802
```