Applied Statistics (Chapter 1)

Yasuhiro Omori

A1A2 term, 2019-2020

University of Tokyo

Chapter 1

Section 1.2

```
# Use ‘LearnBayes’ library (install in advance)
library(LearnBayes)
# Use ‘studentdata’ provided by the author
data(studentdata)
# Show the first row of the data matrix
studentdata[1,]
# Use variable names without ‘studentdata$’
attach(studentdata)
# Show a frequency table for the variable Drink
table(Drink)
# Draw a barplot for Drink
# xlab:label for x axis, ylab:label for y axis
barplot(table(Drink),xlab="Drink",ylab="Count")
```

Section 1.2 (continued)

```
# Open another window for plots (quartz() in Mac)
windows()
# Define a new variable hours.of.sleep
hours.of.sleep = WakeUp - ToSleep
# Output summary statistics for hours.of.sleep
summary(hours.of.sleep)
# Draw a histgram for hours.of.sleep
# main:title of plots
hist(hours.of.sleep,main="")
windows()
# Draw a boxplot for hours.of.sleep by Gender
boxplot(hours.of.sleep~Gender, ylab="Hours of
Sleep")
```

Section 1.2 (continued)

```
# Define a new variable female.Haircut by
# extracting from Haircut if Gender=female
female.Haircut=Haircut[Gender=="female"]
# Define male.Haircut similarly
male.Haircut=Haircut[Gender=="male"]
summary(female.Haircut)
summary(male.Haircut)
windows()
# Draw a scatterplot for ToSleep (x axis) and
# hours.of.sleep (y axis).
# jitter(x):  add some noises to x.
plot(jitter(ToSleep),jitter(hours.of.sleep))
```

Section 1.2 (continued)

```
# Regress hours.of.sleep (y) on ToSleep (x)
# Store the results in fit
fit=lm(hours.of.sleep~ToSleep)
# Show the output
fit
# Add the fitted line to the scatterplot
abline(fit)
# If you want to know more information about lm
# command for linear model, type as follows
help(lm)
```

Section 1.3

```
# Generate X1,..,X10 i.i.d.~ N(50,10^2)
x=rnorm(10,mean=50,sd=10)
y=rnorm(10,mean=50,sd=10)
# length(x):  length of vector x
m=length(x)
n=length(y)
# Compute the square root of the pooled variance
# sd(x):  standard deviation of x
sp=sqrt(((m-1)*sd(x)^2+(n-1)*sd(y)^2)/(m+n-2))
# Compute the t-statistic.  mean(x):mean of x.
t.stat=(mean(x)-mean(y))/(sp*sqrt(1/m+1/n))
```

Section 1.3 (Continued)

```
# Define the function named tstatistic
# x and y are arguments of the function
tstatistic=function(x,y)
# function starts with { and ends with }
{
m=length(x)
n=length(y)
sp=sqrt(((m-1)*sd(x)^2+(n-1)*sd(y)^2)/(m+n-2))
t.stat=(mean(x)-mean(y))/(sp*sqrt(1/m+1/n))
# return the value of t.stat as the output
return(t.stat)
}
```

Section 1.3 (Continued)

```
# Define a vector data.x by combining 1,4,3,6,5
data.x=c(1,4,3,6,5)
data.y=c(5,4,7,6,10)
m=length(x)
n=length(y)
# Output t-statistic using data.x and data.y
tstatistic(data.x, data.y)
# Also output using simulated data, x and y
tstatistic(x, y)
```

Section 1.3 (Continued) Simulation 1

```
# alpha:significance level
# ; denotes the end of commands
# m:length of x, n:length of y.
alpha=0.1; m=10; n=10
# N: the number of simulations
N=10000
# n.reject:counts the number of rejections of H0
n.reject=0
# Repeat the loop from i=1 to i=N
for (i in 1:N)
# the loop starts with { and ends with }
{
x=rnorm(m,mean=0,sd=1)
```

Section 1.3 (Continued) Simulation 1

```
y=rnorm(n,mean=0,sd=1)
t.stat=tstatistic(x,y)
# If |t.stat| > qt, reject H0
# qt:  Pr(T(n+m-2)>qt)=alpha/2
if (abs(t.stat)>qt(1-alpha/2,n+m-2))
n.reject=n.reject+1
}
# Estimate the rejection rate by the relative
# frequency of the rejection and compare
# with alpha =0.1
true.sig.level=n.reject/N
true.sig.level
```

Section 1.3 (Continued) Simulation 2

```
# m:length of x, n:length of y.
m=10; n=10
# Define my.tsimulation as function without
# arguments
my.tsimulation=function()
# When there is only one line, we can omit { }
# rexp:  generate exponential random variables
tstatistic(rnorm(m,mean=10,sd=2),rexp(n,rate=1/10))
# Repeat my.tsimulation 10000 times and save
# t-statistics in tstat.vector
tstat.vector=replicate(10000, my.tsimulation())
```

Section 1.3 (Continued) Simulation 2

```
# Plot the estimated density of tstat.vector
# xlim:  range of x axis, ylim:  range of y axis
# lwd:  line width.  large lwd -> thick lines
plot(density(tstat.vector),xlim=c(-5,8),
ylim=c(0,.4),lwd=3)
# Overwrite (add=TRUE) the density of T(18)
curve(dt(x,df=18),add=TRUE)
```