POLI 272 Bayesian Methods

Homework #1 Answers

1. Exercise 1.4 ***Bayesian Computation With R***

```
#
# Problem 4 Chapter 1 -- Bayesian Computation With R
#
#  Remove all objects just to be safe
#
rm(list=ls(all=TRUE))
#
#  Note -- qnorm(...) is used in the function.  Definitions:

#qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE) -- qnorm gives the quantile
function

#Arguments
#x,q vector of quantiles.
#p vector of probabilities.
#n number of observations. If length(n) > 1, the length is taken to be the number required.
#mean vector of means.
#sd vector of standard deviations.
#log, log.p logical; if TRUE, probabilities p are given as log(p).
#lower.tail logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].

binomial.conf.interval=function(y,n)
{
z=qnorm(.95)
phat=y/n
se=sqrt((phat*(1-phat))/n)
return(c(phat-z*se,phat+z*se))
}
ntrial <- 20
n <- 20
#
confinterval1 <- rep(0,2*ntrial)
dim(confinterval1) <- c(ntrial,2)
confinterval2 <- rep(0,2*ntrial)
dim(confinterval2) <- c(ntrial,2)
#
#Arguments
#n  number of observations. If length(n) > 1, the length is taken to be the number required.
#size      number of trials (zero or more).
#prob      probability of success on each trial.
#rbinom(n, size, prob)

#  Part (b) of the Problem

ptrue <- 0.5

kcover1 <- 0
i <- 1
while (i <= ntrial) {
yarg <- rbinom(1, n, ptrue)
confinterval1[i,] <- binomial.conf.interval(yarg,n)
if (confinterval1[i,1] < ptrue & ptrue < confinterval1[i,2])
     {
      kcover1 = kcover1 + 1
     }
i <- i + 1
}

#  Part (c) of the Problem
```

```
ptrue <- 0.05

kcover2 <- 0
i <- 1
while (i <= ntrial) {
yarg <- rbinom(1, n, ptrue)
confinterval2[i,] <- binomial.conf.interval(yarg,n)
if (confinterval2[i,1] < ptrue & ptrue < confinterval2[i,2])
     {
      kcover2 = kcover2 + 1
     }
i <- i + 1
}
```

b) > confinterval1

```
            [,1]      [,2]
 [1,] 0.53145266 0.8685473
 [2,] 0.41981531 0.7801847
 [3,] 0.31609977 0.6839002
 [4,] 0.47457037 0.8254296
 [5,] 0.36702159 0.7329784
 [6,] 0.17457037 0.5254296
 [7,] 0.53145266 0.8685473
 [8,] 0.41981531 0.7801847
 [9,] 0.31609977 0.6839002
[10,] 0.26702159 0.6329784
[11,] 0.09073773 0.4092623
[12,] 0.36702159 0.7329784
[13,] 0.21981531 0.5801847
[14,] 0.31609977 0.6839002
[15,] 0.41981531 0.7801847
[16,] 0.53145266 0.8685473
[17,] 0.17457037 0.5254296
[18,] 0.17457037 0.5254296
[19,] 0.31609977 0.6839002
[20,] 0.41981531 0.7801847
```

Or 16/20 = 0.8

c) > confinterval2

```
             [,1]      [,2]
 [1,] -0.03016025 0.1301603
 [2,] -0.01034014 0.2103401
 [3,] -0.03016025 0.1301603
 [4,] -0.03016025 0.1301603
 [5,]  0.00000000 0.0000000
 [6,] -0.03016025 0.1301603
 [7,]  0.01866897 0.2813310
 [8,]  0.00000000 0.0000000
 [9,] -0.03016025 0.1301603
```

```
[10,]  0.00000000 0.0000000
[11,] -0.03016025 0.1301603
[12,] -0.03016025 0.1301603
[13,]  0.00000000 0.0000000
[14,]  0.00000000 0.0000000
[15,]  0.01866897 0.2813310
[16,]  0.00000000 0.0000000
[17,]  0.00000000 0.0000000
[18,] -0.03016025 0.1301603
[19,]  0.05287982 0.3471202
[20,] -0.01034014 0.2103401
```

Or 12/20 = 0.6

2. Exercise 1.5 ***Bayesian Computation With R***

I Combined Problems 2 and 3 into one R program (a bit over the top on my part!):

```
#
# Problem 5 Chapter 1 -- Bayesian Computation With R
#
#  Remove all objects just to be safe
#
rm(list=ls(all=TRUE))
library(gdata)
#
#  Note -- qnorm(...) is used in the function.  Definitions:

#qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE) -- qnorm gives the quantile
function

#Arguments
#x,q       vector of quantiles.
#p  vector of probabilities.
#n  number of observations. If length(n) > 1, the length is taken to be the number required.
#mean      vector of means.
#sd        vector of standard deviations.
#log, log.p        logical; if TRUE, probabilities p are given as log(p).
#lower.tail        logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X >
x].
#
#  Confidence Interval for Proportion (Large Sample)
#
binomial.conf.interval=function(y,n)
{
z=qnorm(.95)
phat=y/n
se=sqrt((phat*(1-phat))/n)
return(c(phat-z*se,phat+z*se))
}
#
#  Confidence Interval for Proportion (Quadratic)
#
binomial.conf.interval.quadratic=function(y,n)
{
z <- qnorm(.95)
phat <- y/n
zsquare2 <- (z*z)/n
```

```
lowerc <- ((2*phat + zsquare2) - sqrt((-2*phat-zsquare2)^2 -
4*(1+zsquare2)*phat*phat))/(2*(1+zsquare2))
upperc <- ((2*phat + zsquare2) + sqrt((-2*phat-zsquare2)^2 -
4*(1+zsquare2)*phat*phat))/(2*(1+zsquare2))
return(c(lowerc,upperc))
}
#
#
# Confidence Interval for Proportion Using Large Sample Formula
#
confidence.interval.mc=function(n,p,m)
{
kcover1 <- 0
i <- 1
while (i <= m) {
confinterval1 <- rep(0,2)
dim(confinterval1) <- c(1,2)
yarg <- rbinom(1, n, p)
confinterval1 <- binomial.conf.interval(yarg,n)
if (confinterval1[1] < p & p < confinterval1[2])
     {
      kcover1 = kcover1 + 1
     }
i <- i + 1
}
return(kcover1)
}
#
#  Confidence Interval for Proportion Using Quadratic Formula
#
confidence.interval.quadratic=function(n,p,m)
{
kcover1 <- 0
i <- 1
while (i <= m) {
confinterval1 <- rep(0,2)
dim(confinterval1) <- c(1,2)
yarg <- rbinom(1, n, p)
confinterval1 <- binomial.conf.interval.quadratic(yarg,n)
if (confinterval1[1] < p & p < confinterval1[2])
     {
      kcover1 = kcover1 + 1
     }
i <- i + 1
}
return(kcover1)
}
###############################################
##############  MAIN PROGRAM  ###############
###############################################

confintervalresults <- rep(0,45)
dim(confintervalresults) <- c(9,5)

n <- 10
p <- 0.05
m <- 1000
#
kcovertest <- confidence.interval.mc(n,p,m)
kcovertest2 <- confidence.interval.quadratic(n,p,m)
confintervalresults[1,1] <- n
confintervalresults[1,2] <- p
confintervalresults[1,3] <- m
confintervalresults[1,4] <- kcovertest
confintervalresults[1,5] <- kcovertest2
n <- 10
p <- 0.25
m <- 1000
#
```

```
kcovertest <- confidence.interval.mc(n,p,m)
kcovertest2 <- confidence.interval.quadratic(n,p,m)
confintervalresults[2,1] <- n
confintervalresults[2,2] <- p
confintervalresults[2,3] <- m
confintervalresults[2,4] <- kcovertest
confintervalresults[2,5] <- kcovertest2
n <- 10
p <- 0.50
m <- 1000
#
kcovertest <- confidence.interval.mc(n,p,m)
kcovertest2 <- confidence.interval.quadratic(n,p,m)
confintervalresults[3,1] <- n
confintervalresults[3,2] <- p
confintervalresults[3,3] <- m
confintervalresults[3,4] <- kcovertest
confintervalresults[3,5] <- kcovertest2

n <- 25
p <- 0.05
m <- 1000
#
kcovertest <- confidence.interval.mc(n,p,m)
kcovertest2 <- confidence.interval.quadratic(n,p,m)
confintervalresults[4,1] <- n
confintervalresults[4,2] <- p
confintervalresults[4,3] <- m
confintervalresults[4,4] <- kcovertest
confintervalresults[4,5] <- kcovertest2
n <- 25
p <- 0.25
m <- 1000
#
kcovertest <- confidence.interval.mc(n,p,m)
kcovertest2 <- confidence.interval.quadratic(n,p,m)
confintervalresults[5,1] <- n
confintervalresults[5,2] <- p
confintervalresults[5,3] <- m
confintervalresults[5,4] <- kcovertest
confintervalresults[5,5] <- kcovertest2
n <- 25
p <- 0.50
m <- 1000
#
kcovertest <- confidence.interval.mc(n,p,m)
kcovertest2 <- confidence.interval.quadratic(n,p,m)
confintervalresults[6,1] <- n
confintervalresults[6,2] <- p
confintervalresults[6,3] <- m
confintervalresults[6,4] <- kcovertest
confintervalresults[6,5] <- kcovertest2

n <- 100
p <- 0.05
m <- 1000
#
kcovertest <- confidence.interval.mc(n,p,m)
kcovertest2 <- confidence.interval.quadratic(n,p,m)
confintervalresults[7,1] <- n
confintervalresults[7,2] <- p
confintervalresults[7,3] <- m
confintervalresults[7,4] <- kcovertest
confintervalresults[7,5] <- kcovertest2
n <- 100
p <- 0.25
m <- 1000
#
kcovertest <- confidence.interval.mc(n,p,m)
```

```
kcovertest2 <- confidence.interval.quadratic(n,p,m)
confintervalresults[8,1] <- n
confintervalresults[8,2] <- p
confintervalresults[8,3] <- m
confintervalresults[8,4] <- kcovertest
confintervalresults[8,5] <- kcovertest2
n <- 100
p <- 0.50
m <- 1000
#
kcovertest <- confidence.interval.mc(n,p,m)
kcovertest2 <- confidence.interval.quadratic(n,p,m)
confintervalresults[9,1] <- n
confintervalresults[9,2] <- p
confintervalresults[9,3] <- m
confintervalresults[9,4] <- kcovertest
confintervalresults[9,5] <- kcovertest2
#
write.table(confintervalresults,"c:/docs_Bayesian_statistics/problem_chap_1_5.txt")
write.fwf(x=format(as.data.frame(confintervalresults),digits=5,width=10,
   scientific=FALSE),colnames=FALSE,"c:/docs_Bayesian_statistics/problem_chap_1_5B.txt")
write.fwf(x=format(as.table(confintervalresults),digits=5,width=10,
   scientific=FALSE),colnames=FALSE, "c:/docs_Bayesian_statistics/problem_chap_1_5A.txt")
```

3. **Exercise 1.5 with exact Confidence Limits**

N	P	MC draws	# covered	
			CLT	Exact
10	0.05	1000	383	917
10	0.25	1000	926	860
10	0.50	1000	908	906
25	0.05	1000	714	965
25	0.25	1000	873	897
25	0.50	1000	890	893
100	0.05	1000	865	881
100	0.25	1000	917	930
100	0.50	1000	919	907