Bayesian Inference for Two-Way Contingency Table

Group 6: Xuanyi Li, Xiaoyan Wei, Fangzhou Song

Dataset

```
rm(list=ls())

data=matrix(c(108,19,87,40),2,2,byrow= TRUE)
row.names(data)=c("experimental treatment","control treatment")
colnames(data)=c("success","failure")
data
```

```
##                        success failure
## experimental treatment     108      19
## control treatment           87      40
```

1.Bayesian Methods

As we talked before, we assume the prior distribution of π_1 and π_2 are

$$\pi_1 \sim \text{Beta}(a, b)\pi_2 \sim \text{Beta}(c, d)$$

and let Y_i i=1,2 denote a binomial distribution

$$Y_i \sim \text{Binomial}(n_i, \pi_i)$$

therefore,the posterior distribution of π_1 and π_2 are

$$\pi_1|y_1 \sim \text{Beta}(y_1 + a, n_1 - y_1 + b)\pi_2|y_2 \sim \text{Beta}(y_2 + c, n_2 - y_2 + d)$$

- Standard Bayesian methods: Prior is fixed before any data are observed
- Empirical Bayesian : Let the data suggest hyperparameter values for use in the prior distribution. Maximize the marginal probability of the observed data, integrating out the parameters with respect to that prior
- Hierarchical Bayes:This approach lets the prior hyperparameters themselves have a second-stage prior distribution

---

In this example, we assume that $\pi_1 \sim \text{Beta}(1, 1)$ and $\pi_2 \sim \text{Beta}(1, 1)$
so we have $\pi_1|y_1 \sim \text{Beta}(109, 20)$ and $\pi_2|y_2 \sim \text{Beta}(88, 41)$

```
a=1
b=1
c=1
d=1
y1=data[1,1]
```

```r
n1=sum(data[1,])
y2=data[2,1]
n2=sum(data[2,])
a1=y1+a
b1=n1-y1+b
c1=y2+c
d1=n2-y2+d
```

```
## a1= 109 ,b1= 20 ,c1= 88 ,d1= 41
```

1.1 Simulation of finding $P(\pi_1 > \pi_2)$

```r
n=10000
S1=rbeta(n,a1,b1)
S2=rbeta(n,c1,d1)
sum=0
for(i in 1:n){
  if(S1[i]>=S2[i]){
    sum=sum+1
  }
}
P=sum/n
P
```

```
## [1] 0.9991
```

1.2 Credible Interval of Association Parameters in Bayesian Methods

1.2.1 Difference of Proportions

- Equal Tail

Simulation function

```r
diff.app<- function(a1,b1,c1,d1,conflev,nsim=100000)
  { z1 <- rbeta(nsim, a1,b1)
    z2 <- rbeta(nsim, c1,d1)
    z <- z1 - z2
    z <- sort(z)
    lq <- nsim * (1-conflev)/2
    uq <- nsim * (1 - (1-conflev)/2)
    ci <- array(0,2)
    ci[1] <- z[lq]
    ci[2] <- z[uq]
  return(ci) }
```

Better approximations by integrating the posterior beta densities

```r
fct.F1<- function(x,t,a1,b1,c1,d1){dbeta(x,c1,d1)*pbeta(x+t,a1,b1)}
fct.F2<- function(x,t,a1,b1,c1,d1){dbeta(x,c1,d1)*(1-pbeta(x+t,a1,b1))}

diff.F <- function(t,a1,b1,c1,d1){
```

```
  if(t < 0)
    Fvalue <- integrate(fct.F1,-t,1,t=t,a1=a1,b1=b1,c1=c1,d1=d1)$value
  else
    Fvalue <- 1-integrate(fct.F2,0,1-t,t=t,a1=a1,b1=b1,c1=c1,d1=d1)$value
  return(Fvalue) }

diff.fct <- function(ab,a1,b1,c1,d1,conflev){
  abs(diff.F(ab[2],a1,b1,c1,d1) - (1 - (1-conflev)/2))
  +abs(diff.F(ab[1],a1,b1,c1,d1) - (1-conflev)/2) }

diffCI <- function(x1,n1,x2,n2,a,b,c,d,conflev=.95){
  a1 <- a + x1
  b1 <- b + n1 - x1
  c1 <- c + x2
  d1 <- d + n2 - x2
  start <- diff.app(a1,b1,c1,d1,conflev)
  tailci <- optim(start,diff.fct,a1=a1,b1=b1,c1=c1,d1=d1,
                  conflev=conflev,control=list(maxit=20000))$par
  if(tailci[1] < -1) tailci[1]  <- -1
  if(tailci[2] >  1) tailci[2]  <- 1
  return(tailci)
}
```

Approximate equal tail of Difference of Proportions method using simulation

```
diff.app(a1,b1,c1,d1,0.95)
```

```
## [1] 0.06103113 0.26395110
```

More precise equal tail interval of Difference of Proportions

```
CI_diff_B=diffCI(y1,n1,y2,n2,a,b,c,d)
CI_diff_B
```

```
## [1] 0.06115578 0.28394765
```

- Highest Posterior Density Intervals

```
fct.F1<- function(x,t,a1,b1,c1,d1){
  dbeta(x,c1,d1)*pbeta(x+t,a1,b1)}

fct.F2<- function(x,t,a1,b1,c1,d1){
  dbeta(x,c1,d1)*(1-pbeta(x+t,a1,b1))}

diff.F <- function(t,a1,b1,c1,d1)
{ if(t < 0)
  Fvalue <- integrate(fct.F1,-t,1,t=t,a1=a1,b1=b1,c1=c1,d1=d1)$value
else
  Fvalue <- 1-integrate(fct.F2,0,1-t,t=t,a1=a1,b1=b1,c1=c1,d1=d1)$value
return(Fvalue)
}

fct.f<- function(x,t,a1,b1,c1,d1){
  dbeta(x,c1,d1)*dbeta(x+t,a1,b1)
}

diff.f <- function(t,a1,b1,c1,d1)
```

```
{
  if(t < -1) fvalue <- 100
  else if(t > 1)  fvalue <- 100
  else if((t >= -1) && (t <= 0))
    fvalue <- integrate(fct.f,-t,1,t=t,a1=a1,b1=b1,c1=c1,d1=d1)$value
  else
    fvalue <- integrate(fct.f,0,1-t,t=t,a1=a1,b1=b1,c1=c1,d1=d1)$value
  return(fvalue)
}


diff <- function(ab,a1,b1,c1,d1,conflev)
{
  1000*abs(diff.F(ab[2],a1,b1,c1,d1) -
             diff.F(ab[1],a1,b1,c1,d1) - conflev)+
    abs(diff.f(ab[1],a1,b1,c1,d1) -
          diff.f(ab[2],a1,b1,c1,d1))
}

diffCIhpd <- function(x1,n1,x2,n2,a,b,c,d,conflev=.95)
{
  y <- x1
  if( y > n1/2 ) {
    x2 <- n2-x2
    x1 <- n1-x1
  }
  a1 <- a + x1
  b1 <- b + n1 - x1
  c1 <- c + x2
  d1 <- d + n2 - x2
  start <- diff.app(a1,b1,c1,d1,conflev)
  hdrci <- optim(start,diff,a1=a1,b1=b1,c1=c1,d1=d1,conflev=conflev)$par
  if(hdrci[1] < -1) hdrci[1]  <- -1
  if(hdrci[2] >  1) hdrci[1]  <- 1
  if( y > n1/2 ) {
    ci <- hdrci
    hdrci[1] <- -ci[2]
    hdrci[2] <- -ci[1]
  }
  return(hdrci)
}
```

HPD interval of Difference of Proportions

```
HPD_diff_B=diffCIhpd(y1,n1,y2,n2,a,b,c,d)
HPD_diff_B
```

```
## [1] 0.06134015 0.26418089
```

1.2.2 Relative Risk

- Equal tail

Simulation function

```
risk.app<- function(a1,b1,c1,d1,conflev,nsim=100000)
{
  z1 <- rbeta(nsim, a1,b1)
  z2 <- rbeta(nsim, c1,d1)
  z <- z1/z2
  z <- sort(z)
  lq <- nsim * (1-conflev)/2
  uq <- nsim * (1 - (1-conflev)/2)
  ci <- array(0,2)
  ci[1] <- z[lq]
  ci[2] <- z[uq]
  return(ci)
}
```

Better approximations by integrating the posterior beta densities

```
fct.F1<- function(x,t,a1,b1,a2,b2){
  dbeta(x,a2,b2)*pbeta(x*t,a1,b1)}

fct.F2<- function(x,t,a1,b1,a2,b2){
  dbeta(x,a2,b2)*(1-pbeta(x*t,a1,b1))}

risk.F <- function(t,a1,b1,a2,b2)
{
  if((0<t) && (t<=1)){
    return(integrate(fct.F1,0,1,t=t,a1=a1,b1=b1,a2=a2,b2=b2)$value)
  }
  else{
    return(1-integrate(fct.F2,0,1/t,t=t,a1=a1,b1=b1,a2=a2,b2=b2)$value)
  }
}

risk.fct <- function(ab,a1,b1,c1,d1,conflev)
{
  abs(risk.F(ab[2],a1,b1,c1,d1) - (1 - (1-conflev)/2)) +
    abs(risk.F(ab[1],a1,b1,c1,d1) -(1-conflev)/2)
}

riskCI <- function(x1,n1,x2,n2,a,b,c,d,conflev=.95)
{
  a1 <- a + x1
  b1 <- b + n1 - x1
  c1 <- c + x2
  d1 <- d + n2 - x2
  start <- risk.app(a1,b1,c1,d1,conflev)
  tailci <- optim(start,risk.fct,a1=a1,b1=b1,c1=c1,d1=d1,
                  conflev=conflev,control=list(maxit=20000))$par
  if(tailci[1] < 0) tailci[1]  <- 0
  return(tailci)
}
```

Approximate equal tail of Relative Risk method using simulation

```
risk.app(a1,b1,c1,d1,0.95)
```

```
## [1] 1.083065 1.433045
```

More precise equal tail interval of Relative Risk

```
CI_RR_B=riskCI(y1,n1,y2,n2,a,b,c,d)
CI_RR_B
```

```
## [1] 1.082535 1.432463
```

1.2.3 Odds Ratio

- Equal tail

Simulation

```
or.app<- function(a1,b1,c1,d1,conflev,nsim=1000000)
{
  z1 <- rf(nsim, 2*a1,2*b1)
  z2 <- rf(nsim, 2*c1,2*d1)
  a <- (d1/c1)/(b1/a1)
  z <- a*z1/z2
  z <- sort(z)
  lq <- nsim * (1-conflev)/2
  uq <- nsim * (1 - (1-conflev)/2)
  ci <- array(0,2)
  ci[1] <- z[lq]
  ci[2] <- z[uq]
  return(ci)
}
```

Better approximations by integrating the posterior beta densities

```
fct.F<- function(x,t,a1,b1,a2,b2){
  c <- (b2/a2)/(b1/a1)
  df(x,2*a2,2*b2)*pf(x*t/c,2*a1,2*b1)
}

or.F <- function(t,a1,b1,a2,b2)
{
  return(integrate(fct.F,0,Inf,t=t,a1=a1,b1=b1,a2=a2,b2=b2)$value)
}

or.fct <- function(ab,a1,b1,c1,d1,conflev)
{
  abs(or.F(ab[2],a1,b1,c1,d1) - (1 - (1-conflev)/2))+
    abs(or.F(ab[1],a1,b1,c1,d1) - (1-conflev)/2)
}

orCI <- function(x1,n1,x2,n2,a,b,c,d,conflev=.95)
{
  if(x2!=n2){
    a1 <- a + x1
    b1 <- b + n1 - x1
```

```
    c1 <- c + x2
    d1 <- d + n2 - x2
    start <- or.app(a1,b1,c1,d1,conflev)
    tailci <- optim(start,or.fct,a1=a1,b1=b1,c1=c1,d1=d1,
                    conflev=conflev,control=list(maxit=20000))$par
    if(tailci[1] < 0) tailci[1]  <- 0 }
  else{
    a1 <- a + n1 - x1
    b1 <- b +  x1
    c1 <- c + n2 - x2
    d1 <- d + x2
    start <- or.app(a1,b1,c1,d1,conflev)
    tailci1 <- optim(start,or.fct,a1=a1,b1=b1,c1=c1,d1=d1,
                     conflev=conflev,control=list(maxit=20000))$par
    if(tailci[1] < 0) tailci[1]  <- 0
    tailci <- array(0,2)
    tailci[1] <- 1/ tailci1[2]
    tailci[2] <- 1/ tailci1[1]
  }
  return(tailci)
}
```

Approximate equal tail of Odds Ratio method using simulation

```
or.app(a1,b1,c1,d1,0.95)
```

```
## [1] 1.41450 4.78802
```

More precise equal tail interval of Odds Ratio

```
CI_OR_B=orCI(y1,n1,y2,n2,a,b,c,d)
CI_OR_B
```

```
## [1] 1.413779 4.785396
```

2.Frequentist

2.1 Estimated π_i i=1,2

```
p1_F=data[1,1]/sum(data[1,])
p1_F
```

```
## [1] 0.8503937
```

```
p2_F=data[2,1]/sum(data[2,])
p2_F
```

```
## [1] 0.6850394
```

2.2 Confidence Interval of Association Parameters in Frequentist

2.2.1 Difference of Proportions

```
diff_F=p1_F-p2_F
diff_F
```

```
## [1] 0.1653543
```

```
SE_diff_F=sqrt(p1_F*(1-p1_F)/sum(data[1,])+p2_F*(1-p2_F)/sum(data[2,]))

l=diff_F-qnorm(1-0.05/2)*SE_diff_F
u=diff_F+qnorm(1-0.05/2)*SE_diff_F
CI_diff_F=c(l,u)
CI_diff_F
```

```
## [1] 0.06349904 0.26720962
```

2.2.2 Relative Risk

```
RR_F=p1_F/p2_F
RR_F
```

```
## [1] 1.241379
```

```
logRR_F=log(RR_F)
SE_logRR_F=sqrt((1-p1_F)/data[1,1]+(1-p2_F)/data[2,1])
l=logRR_F-qnorm(1-0.05/2)*SE_logRR_F
u=logRR_F+qnorm(1-0.05/2)*SE_logRR_F
CI_logRR_F=c(l,u)
CI_logRR_F
```

```
## [1] 0.07755679 0.35488943
```

```
CI_RR_F=exp(CI_logRR_F)
CI_RR_F
```

```
## [1] 1.080644 1.426023
```

2.2.3 Odds Ratio

```
OR_F=(data[1,1]*data[2,2])/(data[1,2]*data[2,1])
OR_F
```

```
## [1] 2.61343
```

```
logOR_F=log(OR_F)
logOR_F
```

```
## [1] 0.9606636
```

```
SE_logOR_F=sqrt(sum(1/data))
l=logOR_F-qnorm(1-0.05/2)*SE_logOR_F
u=logOR_F+qnorm(1-0.05/2)*SE_logOR_F
CI_logOR_F=c(l,u)
CI_logOR_F
```

```
## [1] 0.3458935 1.5754337
```

```r
CI_OR_F=exp(CI_logOR_F)
CI_OR_F
```

```
## [1] 1.413252 4.832837
```

---

3. Comparison between Bayesian Methods and Frequentist

3.1 Results Comparison

```r
T1=rbind(CI_diff_B,CI_diff_F)
T2=rbind(CI_RR_B,CI_RR_F)
T3=rbind(CI_OR_B,CI_OR_F)
row.names(T1)=c("Bayesian Method","Frequentist")
row.names(T2)=c("Bayesian Method","Frequentist")
row.names(T3)=c("Bayesian Method","Frequentist")
colnames(T1)=c("Lower Bound","Upper Bound")
colnames(T2)=c("Lower Bound","Upper Bound")
colnames(T3)=c("Lower Bound","Upper Bound")
```

Difference of Proportions

```
##                 Lower Bound Upper Bound
## Bayesian Method  0.06115578   0.2839476
## Frequentist      0.06349904   0.2672096
```

Relative Risk

```
##                 Lower Bound Upper Bound
## Bayesian Method    1.082535    1.432463
## Frequentist        1.080644    1.426023
```

Odds Ratio

```
##                 Lower Bound Upper Bound
## Bayesian Method    1.413779    4.785396
## Frequentist        1.413252    4.832837
```

3.2 Summary

content	Frequentist	Bayesian
Probability is	limiting relative frequency	subjective degree of belief
θ	fixed	random variable
X	random variable	random variable

- Bayesian Goal: Quantify and analyze subjective degrees of belief
- Frequentist Goal: Create procedures that have frequency guarantees

Neither method of inference is right or wrong. Which one you use depends on your goal.If your goal is to quantify and analyze your subjective degrees of belief, you should useBayesian inference. If our goal create procedures that have frequency guarantees, then you should use frequentist procedures.

Sometimes you can do both. That is, sometimes a Bayesian method will also have good frequentist properties. Sometimes it won't.