Homework

# 통계계산입문 과제

## Problem 1 : mammals 자료에 대하여

#### **1 - 1 brain weight/body weight의 비율을 계산하여 새로운 변수로 자료에 추가**

head(mammals)

## body brain  
## Arctic fox 3.385 44.5  
## Owl monkey 0.480 15.5  
## Mountain beaver 1.350 8.1  
## Cow 465.000 423.0  
## Grey wolf 36.330 119.5  
## Goat 27.660 115.0

ratio <- mammals$brain / mammals$body  
mammals$ratio <- ratio  
head(mammals)

## body brain ratio  
## Arctic fox 3.385 44.5 13.1462334  
## Owl monkey 0.480 15.5 32.2916667  
## Mountain beaver 1.350 8.1 6.0000000  
## Cow 465.000 423.0 0.9096774  
## Grey wolf 36.330 119.5 3.2892926  
## Goat 27.660 115.0 4.1576283

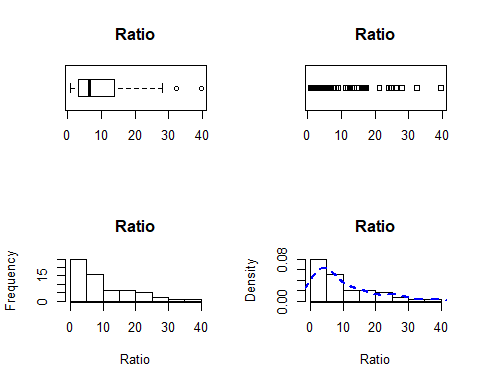
#### **1 - 2 가능한 다양한 그래픽을 통하여 비율자료의 분포 특징을 보여주고**

summary(mammals$ratio)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.8584 3.1026 6.6109 9.6242 13.6684 39.6040

중앙값이 평균보다 낮으니, 왼쪽으로 치우친 그래프라는 것을 알 수 있다.

par(mfrow = c(2, 2))  
boxplot(mammals$ratio, horizontal = T, main = "Ratio")  
stripchart(mammals$ratio, main = "Ratio")  
hist(mammals$ratio, main = "Ratio", xlab = "Ratio")  
hist(mammals$ratio, prob = T, main = "Ratio", xlab = "Ratio")  
lines(density(mammals$ratio), col = "blue", lty = 2, lwd = 2)



par(mfrow = c(1, 1))

#### **1 - 3 folded fraction, folded root, folded log값을 계산하여 [2]를 반복**

mammals$ratio <- mammals$ratio + 0.15   
summary(mammals$ratio)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.008 3.253 6.761 9.774 13.818 39.754

로그를 취했을때 음수가 나오지 않도록 최소값을 1이상으로 만들어준다.

f <- log(mammals$ratio, base = 40)   
summary(f)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.002276 0.319719 0.518084 0.499496 0.711804 0.998327

로그를 취했을 때 값들이 (0, 1)에 위치하도록 base = 40을 설정

ff <- f - (1 - f)   
summary(ff)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -0.995448 -0.360561 0.036167 -0.001008 0.423609 0.996655

값들을 (-1, 1)로 펼쳐줌

froot <- sqrt(2 \* f) - sqrt(2 \* (1 - f))  
summary(froot)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -1.345136 -0.366785 0.036173 -0.009129 0.433976 1.355193

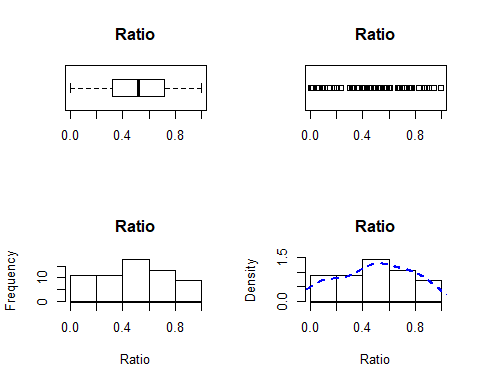
flog <- 1.15 \* log10(f) - 1.15 \* log10(1 - f)  
summary(flog)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -3.03813 -0.37712 0.03614 -0.03380 0.45166 3.19227

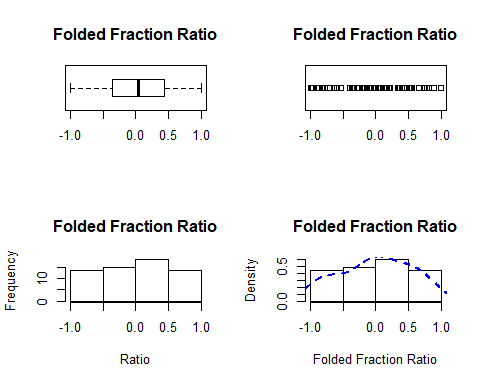
Folded root와 Folded log 값도 0을 기준으로 최대한 대칭을 따르게 만들어주었다.

이제 그래픽 과정을 반복한다.

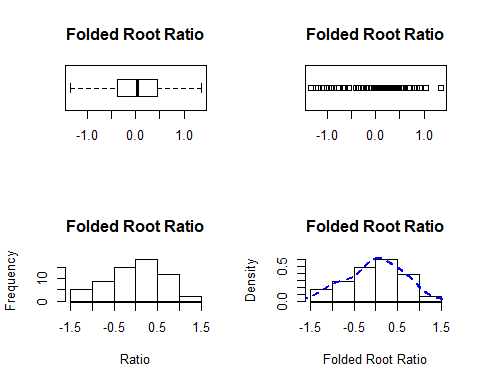
par(mfrow = c(2, 2))  
boxplot(f, horizontal = T, main = "Ratio")  
stripchart(f, main = "Ratio")  
hist(f, main = "Ratio", xlab = "Ratio")  
hist(f, prob = T, main = "Ratio", xlab = "Ratio")  
lines(density(f), col = "blue", lty = 2, lwd = 2)



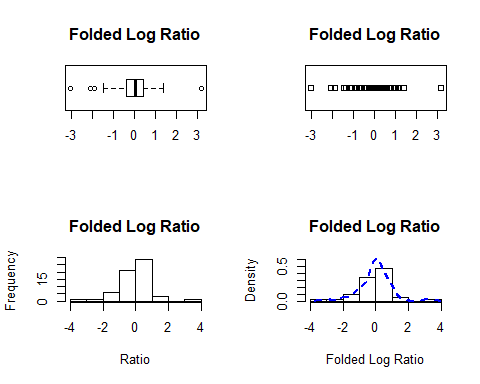
par(mfrow = c(2, 2))  
boxplot(ff, horizontal = T, main = "Folded Fraction Ratio")  
stripchart(ff, main = "Folded Fraction Ratio")  
hist(ff, main = "Folded Fraction Ratio", xlab = "Ratio")  
hist(ff, prob = T, main = "Folded Fraction Ratio", xlab = "Folded Fraction Ratio")  
lines(density(ff), col = "blue", lty = 2, lwd = 2)



par(mfrow = c(2, 2))  
boxplot(froot, horizontal = T, main = "Folded Root Ratio")  
stripchart(froot, main = "Folded Root Ratio")  
hist(froot, main = "Folded Root Ratio", xlab = "Ratio")  
hist(froot, prob = T, main = "Folded Root Ratio", xlab = "Folded Root Ratio")  
lines(density(froot), col = "blue", lty = 2, lwd = 2)



par(mfrow = c(2, 2))  
boxplot(flog, horizontal = T, main = "Folded Log Ratio")  
stripchart(flog, main = "Folded Log Ratio")  
hist(flog, main = "Folded Log Ratio", xlab = "Ratio")  
hist(flog, prob = T, main = "Folded Log Ratio", xlab = "Folded Log Ratio", ylim = c(0, 0.6))  
lines(density(flog), col = "blue", lty = 2, lwd = 2)



#### **1 - 4 비율의 중앙값을 기준으로 큰 집단과 작은 집단으로 분리하는 범주형 변수의 추가**

ifelse 함수를 활용하며 dataframe에 새로운 column 추가

med <- median(mammals$ratio)  
mammals$size <- ifelse(mammals$ratio >= med, "big", "small")  
head(mammals)

## body brain ratio size  
## Arctic fox 3.385 44.5 13.296233 big  
## Owl monkey 0.480 15.5 32.441667 big  
## Mountain beaver 1.350 8.1 6.150000 small  
## Cow 465.000 423.0 1.059677 small  
## Grey wolf 36.330 119.5 3.439293 small  
## Goat 27.660 115.0 4.307628 small

#### **1 - 5 두 집단으로 분리된 자료에 대하여 위의 [2] 및 [3]을 반복**

subset 함수를 이용하여 그룹 나누기

big <- subset(mammals, subset = (size == "big"))  
summary(big$ratio)

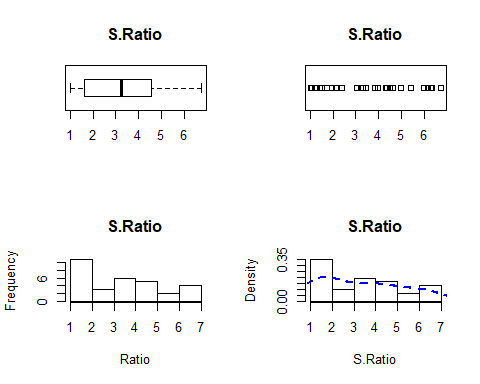
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 6.772 8.832 13.992 16.239 22.814 39.754

sma <- subset(mammals, subset = (size == "small"))  
summary(sma$ratio)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.008 1.610 3.236 3.309 4.561 6.750

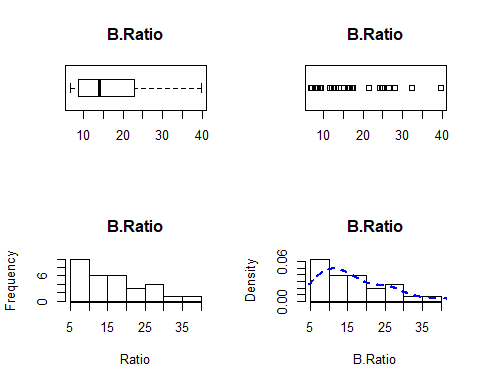
[2]의 반복 for small group

par(mfrow = c(2, 2))  
boxplot(sma$ratio, horizontal = T, main = "S.Ratio")  
stripchart(sma$ratio, main = "S.Ratio")  
hist(sma$ratio, main = "S.Ratio", xlab = "Ratio")  
hist(sma$ratio, prob = T, main = "S.Ratio", xlab = "S.Ratio")  
lines(density(sma$ratio), col = "blue", lty = 2, lwd = 2)



[2]의 반복 for big group

par(mfrow = c(2, 2))  
boxplot(big$ratio, horizontal = T, main = "B.Ratio")  
stripchart(big$ratio, main = "B.Ratio")  
hist(big$ratio, main = "B.Ratio", xlab = "Ratio")  
hist(big$ratio, prob = T, main = "B.Ratio", xlab = "B.Ratio")  
lines(density(big$ratio), col = "blue", lty = 2, lwd = 2)



[3] 반복 for small group

summary(sma$ratio)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.008 1.610 3.236 3.309 4.561 6.750

s <- log(sma$ratio, base = 6.8)  
summary(s)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00438 0.24722 0.61257 0.53402 0.79164 0.99615

sf <- s - (1 - s)  
summary(sf)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -0.99124 -0.50557 0.22514 0.06803 0.58328 0.99230

sroot <- sqrt(2 \* s) - sqrt(2 \* (1 - s))  
summary(sf)

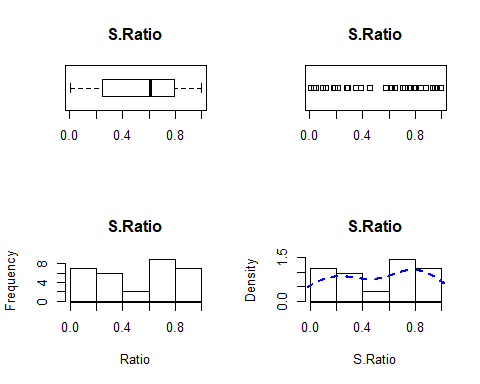
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -0.99124 -0.50557 0.22514 0.06803 0.58328 0.99230

slog <- 1.15 \* log10(s) - 1.15 \* log10(1 - s)  
summary(slog)

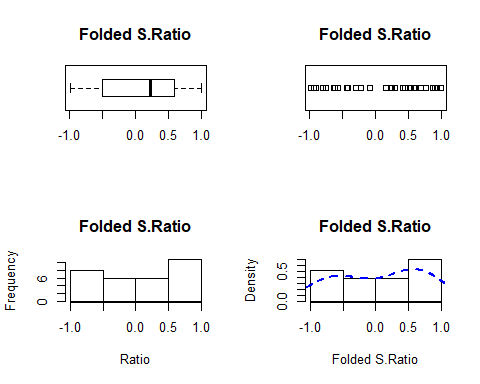
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -2.71014 -0.56083 0.22881 0.09588 0.66683 2.77480

small group에 대한 시각화

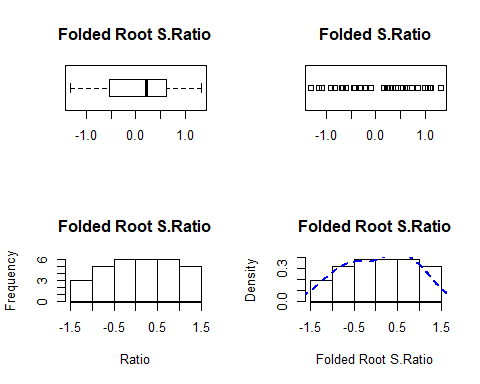
par(mfrow = c(2, 2))  
boxplot(s, horizontal = T, main = "S.Ratio")  
stripchart(s, main = "S.Ratio")  
hist(s, main = "S.Ratio", xlab = "Ratio")  
hist(s, prob = T, main = "S.Ratio", xlab = "S.Ratio")  
lines(density(s), col = "blue", lty = 2, lwd = 2)



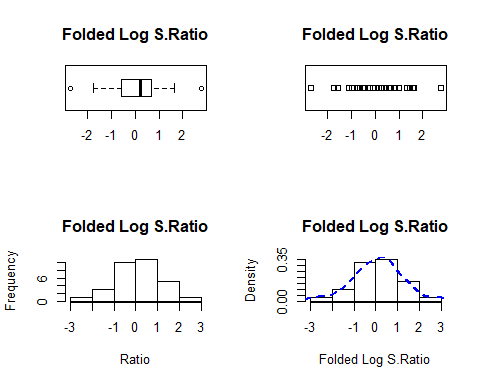
par(mfrow = c(2, 2))  
boxplot(sf, horizontal = T, main = "Folded S.Ratio")  
stripchart(sf, main = "Folded S.Ratio")  
hist(sf, main = "Folded S.Ratio", xlab = "Ratio")  
hist(sf, prob = T, main = "Folded S.Ratio", xlab = "Folded S.Ratio")  
lines(density(sf), col = "blue", lty = 2, lwd = 2)



par(mfrow = c(2, 2))  
boxplot(sroot, horizontal = T, main = "Folded Root S.Ratio")  
stripchart(sroot, main = "Folded S.Ratio")  
hist(sroot, main = "Folded Root S.Ratio", xlab = "Ratio")  
hist(sroot, prob = T, main = "Folded Root S.Ratio", xlab = "Folded Root S.Ratio")  
lines(density(sroot), col = "blue", lty = 2, lwd = 2)



par(mfrow = c(2, 2))  
boxplot(slog, horizontal = T, main = "Folded Log S.Ratio")  
stripchart(slog, main = "Folded Log S.Ratio")  
hist(slog, main = "Folded Log S.Ratio", xlab = "Ratio")  
hist(slog, prob = T, main = "Folded Log S.Ratio", xlab = "Folded Log S.Ratio")  
lines(density(slog), col = "blue", lty = 2, lwd = 2)



[3]의 반복 for big group

big <- mammals[mammals$size == "big", ]  
summary(big$ratio)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 6.772 8.832 13.992 16.239 22.814 39.754

big$ratio <- big$ratio - 5.74  
summary(big$ratio)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.032 3.092 8.252 10.499 17.074 34.014

b <- log(big$ratio, base = 35)  
summary(b)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.008781 0.316609 0.593617 0.549821 0.797202 0.991962

bf <- b - (1 - b)  
summary(bf)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -0.98244 -0.36678 0.18723 0.09964 0.59440 0.98392

broot <- sqrt(2 \* b) - sqrt(2 \* (1 - b))  
summary(broot)

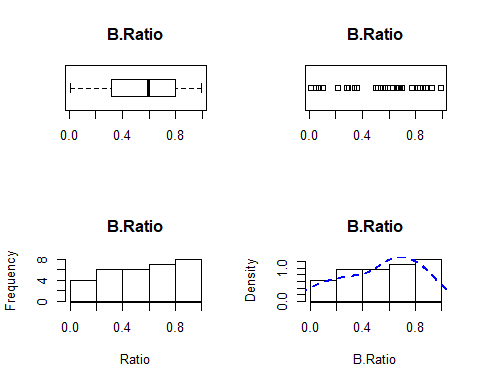
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -1.2755 -0.3737 0.1881 0.1012 0.6267 1.2817

blog <- 1.15 \* log10(b) - 1.15 \* log10(1 - b)  
summary(blog)

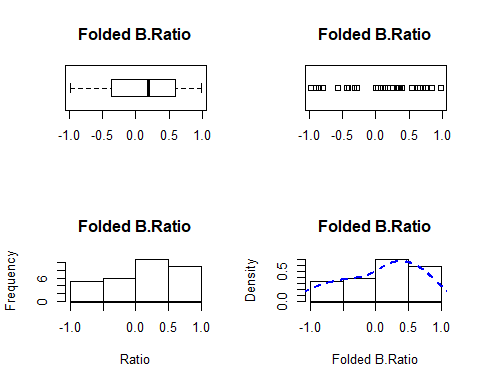
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -2.3605 -0.3853 0.1893 0.1014 0.6866 2.4051

big group에 대한 시각화

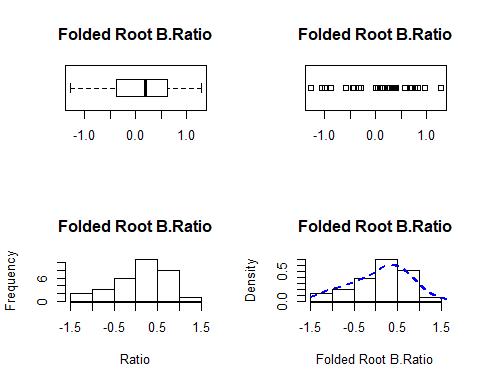
par(mfrow = c(2, 2))  
boxplot(b, horizontal = T, main = "B.Ratio")  
stripchart(b, main = "B.Ratio")  
hist(b, main = "B.Ratio", xlab = "Ratio")  
hist(b, prob = T, main = "B.Ratio", xlab = "B.Ratio")  
lines(density(b), col = "blue", lty = 2, lwd = 2)



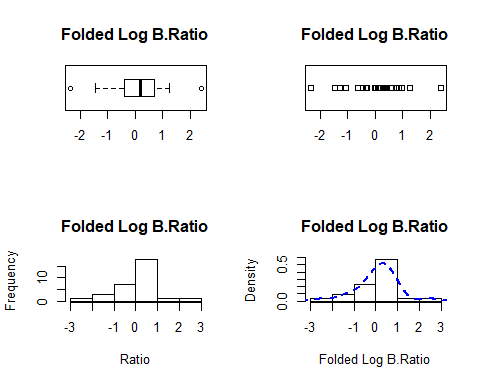
par(mfrow = c(2, 2))  
boxplot(bf, horizontal = T, main = "Folded B.Ratio")  
stripchart(bf, main = "Folded B.Ratio")  
hist(bf, main = "Folded B.Ratio", xlab = "Ratio")  
hist(bf, prob = T, main = "Folded B.Ratio",  
 xlab = "Folded B.Ratio")  
lines(density(bf), col = "blue", lty = 2, lwd = 2)



par(mfrow = c(2, 2))  
boxplot(broot, horizontal = T, main = "Folded Root B.Ratio")  
stripchart(broot, main = "Folded Root B.Ratio")  
hist(broot, main = "Folded Root B.Ratio", xlab = "Ratio")  
hist(broot, prob = T, main = "Folded Root B.Ratio", xlab = "Folded Root B.Ratio")  
lines(density(broot), col = "blue", lty = 2, lwd = 2)



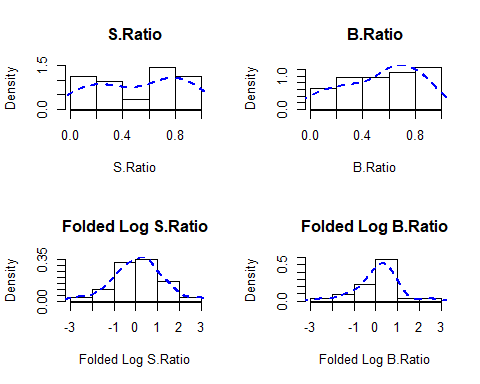
par(mfrow = c(2, 2))  
boxplot(blog, horizontal = T, main = "Folded Log B.Ratio")  
stripchart(blog, main = "Folded Log B.Ratio")  
hist(blog, main = "Folded Log B.Ratio", xlab = "Ratio")  
hist(blog, prob = T, main = "Folded Log B.Ratio", xlab = "Folded Log B.Ratio")  
lines(density(blog), col = "blue", lty = 2, lwd = 2)



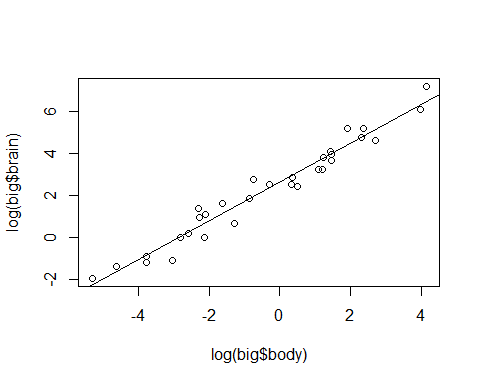
#### **1 - 6 두 집단으로 분리된 각각의 집단별로 body weight와 brain weight간의 관계에**

#### **어떤 차이가 있는지에 대한 정보를 나타내는 그래프 및 분석을 실시**

par(mfrow = c(2, 2))  
hist(s, prob = T, main = "S.Ratio", xlab = "S.Ratio")  
lines(density(s), col = "blue", lty = 2, lwd = 2)  
hist(b, prob = T, main = "B.Ratio", xlab = "B.Ratio")  
lines(density(b), col = "blue", lty = 2, lwd = 2)  
hist(slog, prob = T, main = "Folded Log S.Ratio", xlab = "Folded Log S.Ratio")  
lines(density(slog), col = "blue", lty = 2, lwd = 2)  
hist(blog, prob = T, main = "Folded Log B.Ratio", xlab = "Folded Log B.Ratio")  
lines(density(blog), col = "blue", lty = 2, lwd = 2)

 ratio transformation의 분포를 보았을때는 정확히 어떤지 판단하기 쉽지않다. ratio자체는 small group이나 big group이나 큰 틀에서 고르게 분포한다고 볼 수도 있지만, 정확히 말하기 어렵다. log ratio는 값들을 중앙으로 모아주는 성질때문에, 정규분포처럼 보이지만 적절한 결과를 도출하기 어렵다. 따라서 각 그룹의 brain weight와 body weight를 축으로 하는 그래프를 그리려한다.

big.fit <- lm(log(big$brain)~log(big$body))  
plot(log(big$body), log(big$brain))  
abline(big.fit)

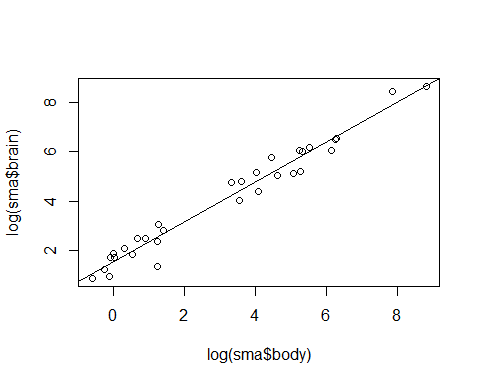
 big group의 경우, 값들이 회귀선을 중심으로 고르게 퍼져있다는 점을 확인할 수 있고, 둘 사이에 양의 상관관계를 확인할 수 있다.

cor.test(log(big$body), log(big$brain))

##   
## Pearson's product-moment correlation  
##   
## data: log(big$body) and log(big$brain)  
## t = 25.826, df = 29, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.9563384 0.9899065  
## sample estimates:  
## cor   
## 0.9789445

이를 바탕으로 상관분석을 시행한 결과 상관계수는 0.978로, 두 변수는 매우 큰 상관도를 지닌다.

sma.fit <- lm(log(sma$brain)~log(sma$body))  
plot(log(sma$body), log(sma$brain))  
abline(sma.fit)

 small group도 big group과 마찬가지로 회귀선을 중심으로 고르게 분포되어 있다. 하지만 중간에 공백이 있는것으로 보아, 두 그룹보다는 세 그룹으로 나누는 것을 생각해봐야 할 수도 있겠다.

cor.test(log(sma$body), log(sma$brain))

##   
## Pearson's product-moment correlation  
##   
## data: log(sma$body) and log(sma$brain)  
## t = 28.188, df = 29, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.9630968 0.9914915  
## sample estimates:  
## cor   
## 0.9822355

small group도 상관계수가 0.982로 매우 높다

##### **body weight와 brain weight는 매우 큰 양의 상관관계를 지닌다.**

## Problem 2

#### **Exercise 1.14 (Tearing factor of paper)**

tf <- read.table("tearingfactor.txt", header = T)  
summary(tf)

## Pressure TearFactor   
## Min. : 35.0 Min. : 96.0   
## 1st Qu.: 49.5 1st Qu.:101.0   
## Median : 70.0 Median :107.0   
## Mean : 78.7 Mean :107.4   
## 3rd Qu.: 99.0 3rd Qu.:112.2   
## Max. :140.0 Max. :120.0

## Problem 3

####**Exercise 2.14 and 2.15 (clustering and grouping)**

**for 2.14 mammals cluster**

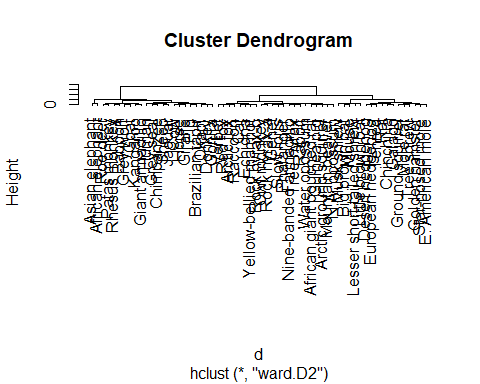
head(mammals)

## body brain  
## Arctic fox 3.385 44.5  
## Owl monkey 0.480 15.5  
## Mountain beaver 1.350 8.1  
## Cow 465.000 423.0  
## Grey wolf 36.330 119.5  
## Goat 27.660 115.0

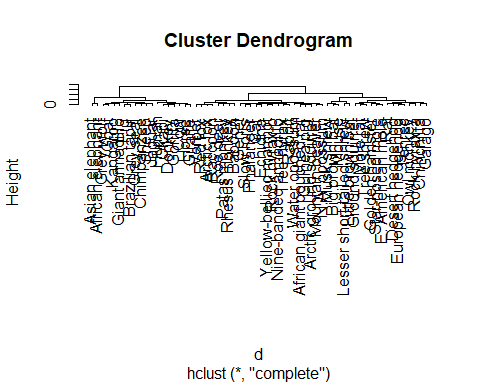
summary(mammals)

## body brain   
## Min. : 0.005 Min. : 0.14   
## 1st Qu.: 0.600 1st Qu.: 4.25   
## Median : 3.342 Median : 17.25   
## Mean : 198.790 Mean : 283.13   
## 3rd Qu.: 48.203 3rd Qu.: 166.00   
## Max. :6654.000 Max. :5712.00

d <- dist(log(mammals))  
h <- hclust(d, method = "ward.D2")  
plot(h)



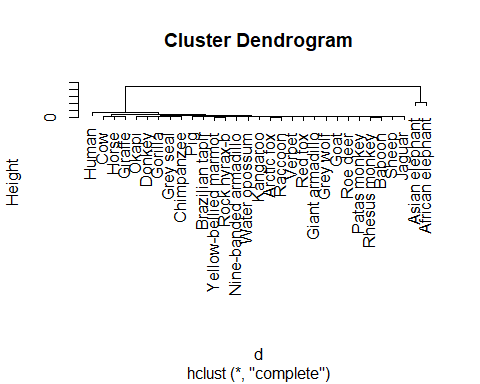
h <- hclust(d, method = "complete")  
plot(h)



**for 2.15 identifying groups or cluster**

big <- subset(mammals, subset = (body > median(body)))

d <- dist(big)  
h <- hclust(d, method = "complete")  
plot(h)

 **Display label of each observation**

g <- cutree(h,5)  
labels(g)

## [1] "Arctic fox" "Cow"   
## [3] "Grey wolf" "Goat"   
## [5] "Roe deer" "Verbet"   
## [7] "Nine-banded armadillo" "Asian elephant"   
## [9] "Donkey" "Horse"   
## [11] "Patas monkey" "Giraffe"   
## [13] "Gorilla" "Grey seal"   
## [15] "Human" "African elephant"   
## [17] "Water opossum" "Rhesus monkey"   
## [19] "Kangaroo" "Yellow-bellied marmot"  
## [21] "Okapi" "Sheep"   
## [23] "Jaguar" "Chimpanzee"   
## [25] "Baboon" "Giant armadillo"   
## [27] "Rock hyrax-b" "Raccoon"   
## [29] "Pig" "Brazilian tapir"   
## [31] "Red fox"

**summarize group size**

table(g)

## g  
## 1 2 3 4 5   
## 25 3 1 1 1

big[g > 2, ]

## body brain  
## Asian elephant 2547 4603  
## Human 62 1320  
## African elephant 6654 5712

## Problem 4

#### **Exercise 3.3 (Does baseball hitting data follow a binomial distribution?)**

**for 3.3 Does baseball hitting data follow a binomial distribution** (1)

k <- 0 : 4  
p <- dbinom(k, 4, 0.312)  
binom <- round(70 \* p)  
binom

## [1] 16 28 19 6 1

names(binom) = k  
binom

## 0 1 2 3 4   
## 16 28 19 6 1

pujols <- c(17, 31, 17, 5)  
pujols

## [1] 17 31 17 5

prob <- c(p[1:3], 1 - sum(p[1:3]))  
prob

## [1] 0.22405454 0.40642452 0.27646319 0.09305775

chisq.test(pujols, p = prob)

##   
## Chi-squared test for given probabilities  
##   
## data: pujols  
## X-squared = 0.97692, df = 3, p-value = 0.8068

H0는 차이가 없다는 것인데, 이를 기각할 수 없다.

k <- 0 : 5  
p <- dbinom(k, 5, 0.312)  
binom <- round(25 \* p)  
binom

## [1] 4 9 8 4 1 0

names(binom) = k  
binom

## 0 1 2 3 4 5   
## 4 9 8 4 1 0

pujols <- c(5, 5, 4, 11)  
pujols

## [1] 5 5 4 11

sum(p[1:3])

## [1] 0.8206857

prob <- c(p[1:3], 1 - sum(p[1:3]))  
prob

## [1] 0.1541495 0.3495251 0.3170111 0.1793143

chisq.test(pujols, p = prob)

## Warning in chisq.test(pujols, p = prob): Chi-squared approximation may be  
## incorrect

##   
## Chi-squared test for given probabilities  
##   
## data: pujols  
## X-squared = 13.359, df = 3, p-value = 0.003922

pvalue가 매우 낮기때문에, 푸홀스의 타율과 이항분포의 확률에는 차이가 있다.

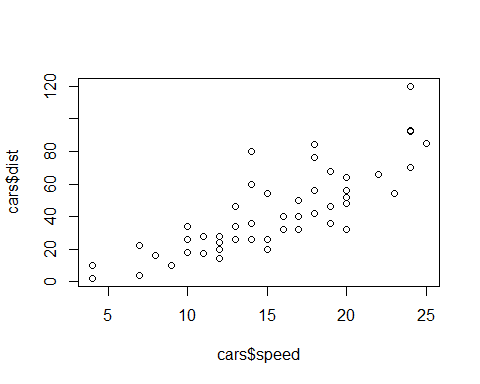
## Problem 5

#### **Exercise 4.1 and 4.2 (Speed and stopping distance)**

##### **4.1 (Speed and stopping distance).**

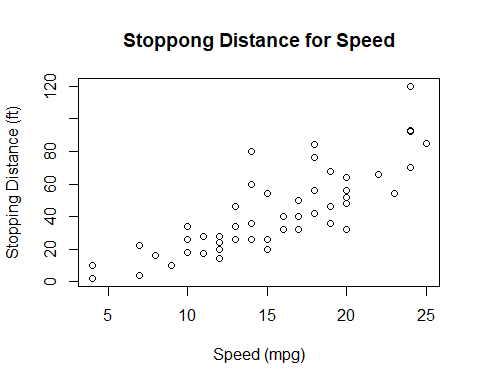
**a** Use the plot function to construct a scatterplot of speed (horizontal) against dist (vertical).

plot(cars$speed, cars$dist)



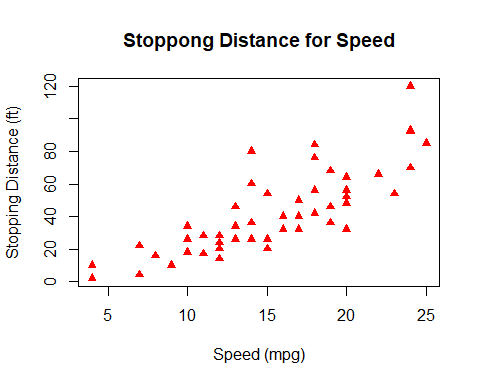
**b** Revise the basic plot by labeling the horizontal axis with “Speed (mpg)” and the vertical axis with “Stopping Distance (ft),” Add a meaningful title to the plot.

plot(cars$speed, cars$dist, xlab = "Speed (mpg)", ylab = "Stopping Distance (ft)",  
 main = "Stoppong Distance for Speed")



**c** Revise the plot by changing the plot symbol from the default open circles to red filled triangles (col=“red”, pch=17).

plot(cars$speed, cars$dist, xlab = "Speed (mpg)", ylab = "Stopping Distance (ft)",  
 main = "Stoppong Distance for Speed", col = "red", pch = 17)



#### **4.2 (Speed and stopping distance (continued)).**

fit.linear = lm(dist ~ speed, data=cars)  
fit.linear

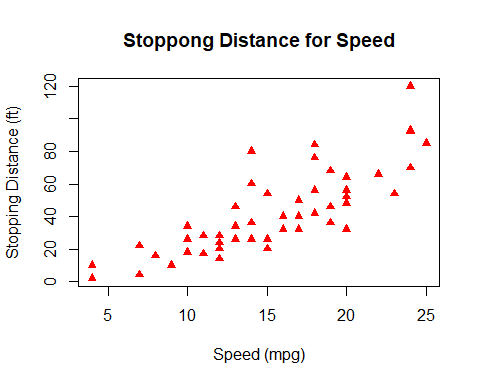
##   
## Call:  
## lm(formula = dist ~ speed, data = cars)  
##   
## Coefficients:  
## (Intercept) speed   
## -17.579 3.932

fit.quadratic = lm(dist ~ speed + I(speed^2), data=cars)  
fit.quadratic

##   
## Call:  
## lm(formula = dist ~ speed + I(speed^2), data = cars)  
##   
## Coefficients:  
## (Intercept) speed I(speed^2)   
## 2.47014 0.91329 0.09996

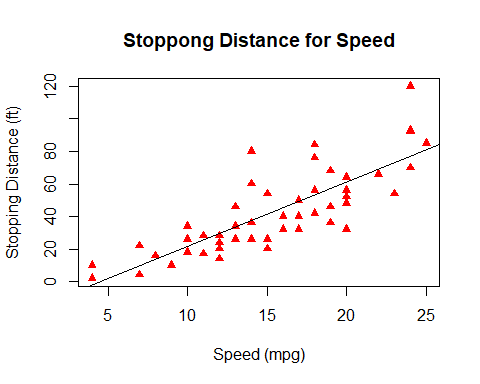
**a.** Construct a scatterplot of speed and stopping distance.

plot(cars$speed, cars$dis, xlab = "Speed (mpg)", ylab = "Stopping Distance (ft)",  
 main = "Stoppong Distance for Speed", col = "red", pch = 17)



**b.** Using the abline function with argument fit.linear, overlay the best line fit using line type “dotted” and using a line width of 2.

plot(cars$speed, cars$dist, xlab = "Speed (mpg)", ylab = "Stopping Distance (ft)",  
 main = "Stoppong Distance for Speed", col = "red", pch = 17)  
abline(fit.linear)

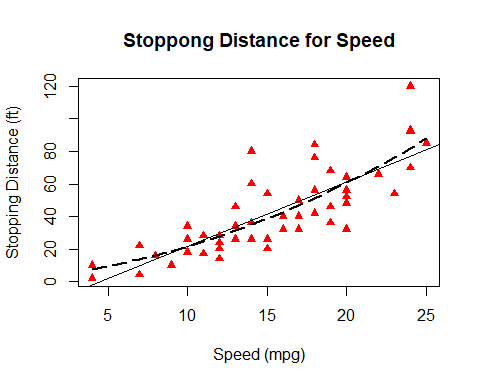


**c.** Using the lines function, overlay the quadratic fit using line type “longdash” and a line width of 2.

attach(cars)  
fit.quadratic

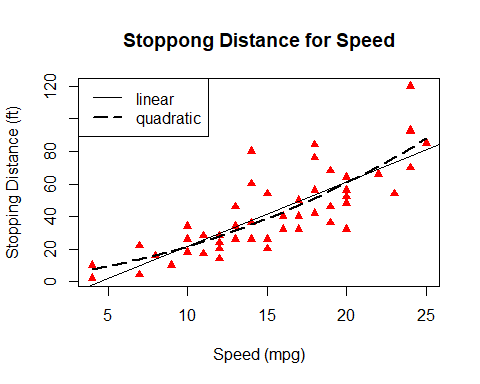
##   
## Call:  
## lm(formula = dist ~ speed + I(speed^2), data = cars)  
##   
## Coefficients:  
## (Intercept) speed I(speed^2)   
## 2.47014 0.91329 0.09996

plot(cars$speed, cars$dist, xlab = "Speed (mpg)", ylab = "Stopping Distance (ft)",  
 main = "Stoppong Distance for Speed", col = "red", pch = 17)  
abline(fit.linear)  
lines(speed, fit.quadratic$fitted.values, lty = 5 ,lwd = 2)



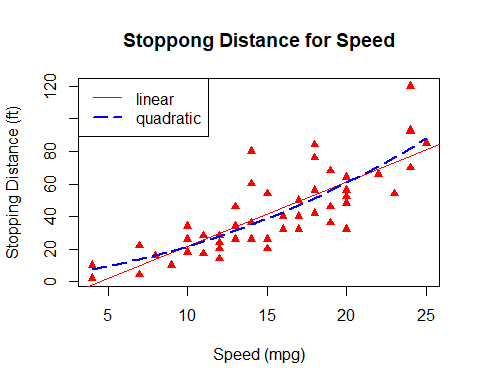
**d.** Use a legend to show the line types of the linear and quadratic fits.

plot(cars$speed, cars$dist, xlab = "Speed (mpg)", ylab = "Stopping Distance (ft)",  
 main = "Stoppong Distance for Speed", col = "red", pch = 17)  
abline(fit.linear)  
lines(speed, fit.quadratic$fitted.values, lty = 5 ,lwd = 2)  
legend("topleft", legend = c("linear", "quadratic"), lty = c(1, 5), lwd = c(1, 2))



**e.** Redo parts (a) - (d) using two contrasting colors (say red and blue) for the two different fits.

plot(cars$speed, cars$dist,xlab = "Speed (mpg)", ylab = "Stopping Distance (ft)",  
 main = "Stoppong Distance for Speed", col = "red", pch = 17)  
abline(fit.linear, col = "red")  
lines(speed, fit.quadratic$fitted.values, lty = 5 ,lwd = 2, col = 'blue')  
legend("topleft", legend = c("linear", "quadratic"), lty = c(1, 5), lwd = c(1, 2), col = c("red", "blue"))



detach(cars)

## Problem 6

1. 아래 제시된 내용을 모두 수행하는 R 함수 (function)을 작성하고 임의로 선택된 k개 점들에 대하여 수행한 결과를 프로그램소스와 함께 정리하여 제출하시오. [1] 함수의 인수는 k (자연수) [1] 적당한 수평축과 수직축의 범위를 갖는 새로운 플롯창을 오픈 [2] 임의의 k개 점을 선택하여 좌표를 저장 [3] 선택된 k개의 좌표에 적당한 point character를 사용하여 표시 [4] k개의 점을 시작점에서 시작하여 끝점까지 화살표로 연결 [5] 플롯창에 테두리를 추가

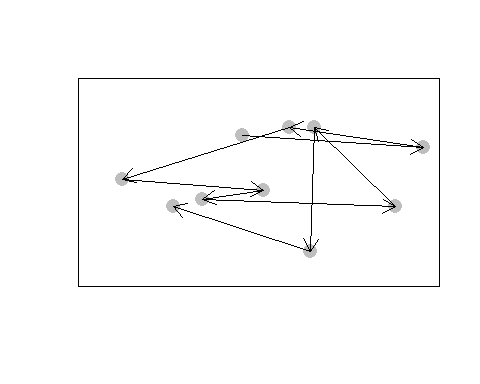
**locator 함수를 이용해서, 좌표에 점을 찍을수 있다.** 함수의 정의는 다음과 같다.

gbl = function(k){  
 plot(NULL, xaxt = "n", yaxt = "n", bty = "n", ylab = "", xlab = "",  
 xlim = c(0, 1), ylim = c(0, 1))  
 coord = locator(k, type = "p", pch = 16)  
 for(n in 1 : (k - 1)) {  
 arrows(coord$x[1], coord$y[1], coord$x[n + 1], coord$y[n + 1],   
 length = 0.15)  
 }  
}  
gbl(5)

하지만 이는 R markdown에선 실행이 안되기때문에, 코드만 작성했다. 이와 유사하게 locator 함수 없이 runif함수를 통해 좌표를 생성하겠다.

gbl = function(k){  
 plot(NULL, xaxt = "n", yaxt = "n", bty = "n", ylab = "", xlab = "",  
 xlim = c(0, 1), ylim = c(0, 1))  
 xcor = runif(k, 0, 1)  
 ycor = runif(k, 0, 1)  
 coord = cbind(xcor, ycor)  
 points(coord, cex = 2, pch = 16, col = "gray")  
 n = 1  
 while (n < k){  
 arrows(coord[n, 1], coord[n, 2], coord[n + 1, 1], coord[n + 1, 2], length = 0.15)  
 n = n + 1  
 }  
 box()  
}

gbl(10)

 이와 같이 생성이 가능하다.

## Problem 7.

Website counts 자료(10.4절 및 10.5절)에 대한 permutation test의 내용 중 correlation의 분포를 이용하는 부분을 correlation test를 이용하는 내용에 적합하도록 교재의 해당 프로그램들을 모두 수정하고 각자 작성한 프로그램과 결과를 정리하여 제출하시오.

**책에 주어진 permutation test 내용**

rand.correlation = function(x, y, R=199) {  
 ranks.x = rank(x)  
 ranks.y = rank(y)  
 observed.r = cor(ranks.x, ranks.y)  
 stats = replicate(R, {  
 random.ranks = sample(ranks.y)  
 cor(ranks.x, random.ranks)  
 })  
 p.value = sum(stats >= observed.r) / (R + 1)  
 list(observed.r = observed.r, p.value = p.value)  
}

cor(rank.x, rank.y)와 같은 부분을, cor.test로 바꿔야한다. 하지만 이때 cor.test를 할 경우, pearson correlation test로 진행된다. permutatatin test는 비모수적 방법이기때문에, pearson방법 말고 spearman rank correlation을 이용해야한다. 또한 이때 tie들로 인해, 정확한 확률이 구해지지 않는다는 경고메시지가 발생하는데, exact = F를 추가해주면 이를 억제할 수 있다.

rand.correlation4 = function(x, y, R=199) {  
 ranks.x = rank(x)  
 ranks.y = rank(y)  
 observed.r = cor.test(ranks.x, ranks.y, method = "spearman", exact = F)$estimate  
 stats = replicate(R, {  
 random.ranks = sample(ranks.y)  
 cor.test(ranks.x, random.ranks, method = "spearman", exact = F)$estimate  
 })  
 p.value = sum(stats >= observed.r) / (R + 1)  
 list(observed.r = observed.r, p.value = p.value)  
}

rand.correlation4(web$Week, web$Hits, R=100000)

## $observed.r  
## rho   
## 0.3217489   
##   
## $p.value  
## [1] 0.0298097

p-value는 0.0965로, 유의수준이 0.05였다면 rho = 0 이라는 귀무가설을 기각한다.

## Problem 8.

#### A coin-tossing game (11.2절)의 내용을 수정하여 Peter가 60불을 가지고 있고 Paul은

#### 40불을 가지고 있으며 게임의 룰은 두 사람 중 한 사람의 돈이 처음으로 0이 되는 시점

#### 에 끝나는 것으로 할 때

**[1] Peter가 이길 확률**

p.peter.win = function(peter=60){  
 repeat  
 {  
 peter = peter + sample(c(-1, 1), 1)  
 if (peter == 100){  
 return("Win")  
 break}  
 if (peter == 0){  
 return("Lose")  
 break}  
 }  
}  
p.peter.win()

## [1] "Win"

피터가 이기는 경우는 “Win”를 반환하고, 지는경우 “Lose”를 반환하는 함수를 작성

n.win = replicate(1000, p.peter.win())

이 과정을 1000번 반복했다

a = prop.table(table(n.win))  
a["Win"]

## Win   
## 0.612

직관적으로 0.60 정도가 나올것 같았고, 시뮬레이션 결과 0.569가 나왔다.

**[2] 게임이 끝날 때 까지의 평균시간(동전을 던지는 횟수)**

n.game = function(peter = 60){  
 n=0  
 while (peter > 0 & peter < 100)  
 {  
 peter = peter + sample(c(-1, 1), 1)  
 n = n + 1  
 }  
 return(n)  
}

게임이 끝날 때 까지의 횟수를 계산하는 함수를 만들었다.

mean(replicate(1000, n.game()))

## [1] 2307.602

이를 1000번 반복해서 평균을 구한 결과, 2400.836이 나왔다. 평균적으로 2400번 정도에서 게임은 종료될 것이다.

## Problem 9

#### **Exercise 11.2(Checking hats)**

**a** Modify the function scramble.hats to compute the number of correct matches in this setting. (The only change is the definition of the vector hats – if one represents a black hat and a grey hat by 1 and 2, respectively, then hats consists of ten 1’s and ten 2’s.)

scramble.hats = function() {  
 hats = rep(c(1, 2), each = 10)  
 mixed.hats <- sample(hats)  
 correct <- sum(hats == mixed.hats)  
 return(correct)  
}

몇 명이 바르게 쓰고 나가는지를 계산하는 함수를 작성했다.

scramble.hats()

## [1] 10

**b** Using the function replicate, repeat this simulation for 1000 trials. Store the number of matches for the 1000 experiments in the vector matches.

matches <- replicate(1000, scramble.hats())

이를 1000번 반복했다.

**c** From the simulated values, approximate the probability that 10 or more men receive the correct hats. Also, find the expected number of correct matches.

sum(matches >= 10) / length(matches) #approximate probability

## [1] 0.648

mean(matches) #expected number of correct matches

## [1] 9.864

같은 모자를 찾아갈 확률은 0.681로 계산이 되고, 평균적으로 같은 모자를 가져가는 사람의 수는 10.032이다.