FIT5197 2018 S1 Assignment 1

KEWEN DENG, 29330440

13 April 2018

Question 1: calculate conditional probability of an event

And in this question, p(B) can be calculated as

$$
\begin{aligned}
p(\mathrm{B}) &= (\frac{6*5*5*5*5*5*5}{6*6*6*6*6*6*6}) \\
&= \frac{5^6*6}{6^7}
\end{aligned}
$$

in which, 6 * 5 * 5 * 5 * 5 * 5 * 5 means the total times fitting the condition of event B, and 6 * 6 * 6 * 6 * 6 * 6 * 6 means the totals times of all the occurrences could happen.

Also, p(AB) can be calculated as

$$
\begin{aligned}
p(\mathrm{A}\bigcap B) &= (\frac{6*5*4*3*2*1*C_2^6}{6*6*6*6*6*6*6}) \\
&= \frac{6!*15}{6^7}
\end{aligned}
$$

in which, 6 * 5 * 5 * 5 * 5 * 5 * 5 means the total times fitting the condition of both event A and event B, and 6 * 6 * 6 * 6 * 6 * 6 * 6 means the same in p(B).

Consequently, the conditional probability can be calculated by

$$
\begin{aligned}
p(\mathrm{A}|\mathrm{B}) &= \frac{p(\mathrm{A}\bigcap B)}{p(\mathrm{B})} \\
&= \frac{\frac{6!*15}{6^7}}{\frac{5^6*6}{6^7}} = \frac{6!*15}{5^6*6} = \frac{10800}{93750} \\
&\approx 0.115
\end{aligned}
$$

Simulation

To solve this problem using simulation, we run the 7 times fair die tossing process n times. If the analytical solution was correct, the simulation result should converge to it as $n \to \infty$. Below are the R codes for simulation.

```
# Judge whether the outcome is alternate in numbers or not.
judge_ad <- function(sevenTosses) {
  bool = T
  for (j in 1:6){
    # If adjacent, returns FALSE back and vice versa.
    if(sevenTosses[j] == sevenTosses[j+1]){bool = F}
  }
```

```r
  return(bool)
}

# Judge whether each value appears at least once or not.
judge_one <- function(sevenTosses) {
  # If appears at least once, returns TRUE and vice versa.
  all(sides%in%sevenTosses)
}

# main
nRuns = 1000000 # Simulation times
cntAd = 0 # Initialization
cntOne = 0 # Initialization
sides = c(1:6) # Set 6 sizes of the fair die

for (i in 1:nRuns){ # Simulate for nRuns times
  # Generate 7 values from 1 to 6, which represents tossing fair die for 7 times.
  sevenTosses = sample(c(1:6), 7, replace = T)
  if(judge_ad(sevenTosses)){ # Judge event B
    if(judge_one(sevenTosses)){# Judge event A when it belongs to event B
      cntOne = cntOne + 1 # Count the occurrences of event A and event B happened together.
    }
    cntAd = cntAd + 1 # Count the occurrences of event B happened.
  }
}

# Get the conditional probability
# Use the count of both event A and event B happened to divide the count of event B happened.
(p = cntOne/cntAd)
```

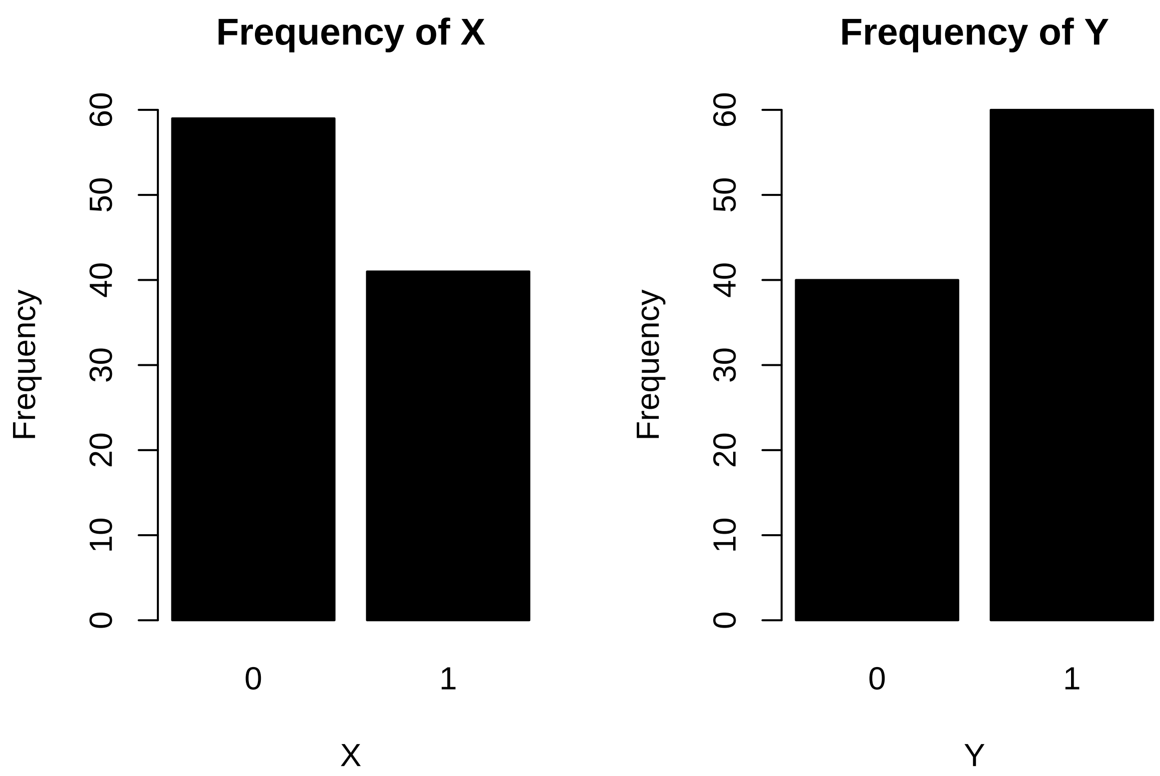
```
## [1] 0.11433
```

Question 2: entropy

Q2.1 Imputation and Plot

According to the question, below are the R codes for handling NAs by mode imputation and plotting individual variables in barplots.

```
# Initialization
data.path = 'D:/Desktop/5197-Assignment1'
setwd(data.path)
data.file = c('FIT5197_2018_S1_Assignment1_Q2_data.csv')
rawdata = read.csv(data.file)

# Mode
get_mode <- function(x){ # Define a function to get the mode easily
  return(as.numeric(names(table(x)[table(x)==max(table(x))])))
}

# # Imputate missing value with mode number
rawdata$X[is.na(rawdata$X)]<- get_mode(rawdata$X)
rawdata$Y[is.na(rawdata$Y)]<- get_mode(rawdata$Y)

# Barplot
par(mfrow = c(1,2)) # Make a 1*2 place for plot
barplot(table(rawdata$X), # Use barplot to show the  frequency of X
        main = 'Frequency of X',
        xlab = 'X',
        ylab = 'Frequency',
        ylim = c(0,60),
        col = "black")

barplot(table(rawdata$Y), # Use barplot to show the  frequency of Y
        main = 'Frequency of Y',
        xlab = 'Y',
        ylab = 'Frequency',
        ylim = c(0,60),
        col = "black")
```



Q2.2 Full tables for Probability

After imputation, $p(\mathrm{X})$ can be calculated as

$$p(\mathrm{X}{=}1) = \frac{41}{100} = 0.41$$
$$p(\mathrm{X}{=}0) = 1 - p(\mathrm{X}{=}1) = 0.59$$

Below are the R codes for calculation.

```
print(prop.table(table(rawdata$X)))
```

```
##
##    0    1
## 0.59 0.41
```

And, $p(\mathrm{Y})$ can becalculated as

$$p(\mathrm{Y}{=}1) = \frac{60}{100} = 0.60$$
$$p(\mathrm{Y}{=}0) = 1 - p(\mathrm{Y}{=}1) = 0.40$$

Below are the R codes for calculation.

```r
print(prop.table(table(rawdata$Y)))
```

```
##
##   0   1
## 0.4 0.6
```

Also, the full table of p(X,Y) is

$$p(\mathrm{X{=}1,\ Y{=}1}) = p(\mathrm{X{=}1}) * p(\mathrm{Y{=}1}) = 0.41 * 0.60 = 0.246$$
$$p(\mathrm{X{=}0,\ Y{=}1}) = p(\mathrm{X{=}0}) * p(\mathrm{Y{=}1}) = 0.59 * 0.60 = 0.354$$
$$p(\mathrm{X{=}1,\ Y{=}0}) = p(\mathrm{X{=}1}) * p(\mathrm{Y{=}0}) = 0.41 * 0.40 = 0.164$$
$$p(\mathrm{X{=}0,\ Y{=}0}) = p(\mathrm{X{=}0}) * p(\mathrm{Y{=}0}) = 0.59 * 0.40 = 0.236$$

Below are the R codes for calculation.

```r
print(prop.table(table(rawdata)))
```

```
##    Y
## X      0    1
##   0 0.23 0.36
##   1 0.17 0.24
```

Moreover, $p(\mathrm{X}\bigcap Y)$ can be counted and calculated, which is

$$p(\mathrm{X}\bigcap Y) = \frac{24}{100} = 0.24$$

Thus, p(X|Y) and p(Y|X) can be calculated as

$$p(\mathrm{X|Y}) = \frac{p(\mathrm{X}\bigcap Y)}{p(\mathrm{Y})} = 0.4$$
$$p(\mathrm{Y|X}) = \frac{p(\mathrm{X}\bigcap Y)}{p(\mathrm{X})} \approx 0.585$$

Below are the R codes for calculation.

```r
cntX = 0 # Initialization, count of X
cntY = 0 # Initialization, count of Y
cntXY = 0 # Initialization,count of X and Y are '1' together
n = length(rawdata$X) # The amount of X, and is also the amount of Y is this case.

for (i in 1:n){ # Count probabilities of (X,Y) from the first to the last.
  # if X=1, count of X plus 1
  if(rawdata[i,1] == '1'){cntX = cntX + 1}
  # if Y=1, count of Y plus 1
  if(rawdata[i,2] == '1'){cntY = cntY + 1}
  # if X=1 and Y=1, count of X and Y plus 1
  if(rawdata[i,1] == '1' & rawdata[i,2] == '1'){cntXY = cntXY + 1}
}
cat(' P(X|Y) =',cntXY/cntY,'\n',
    'P(Y|X) =',cntXY/cntX)
```

```
##  P(X|Y) = 0.4
##  P(Y|X) = 0.5853659
```

Q2.3 Single values for Entropy

Moreover, the entropy function can be calculated by using

$$H(\text{p}) = \sum_{i=i}^{K} p_i log_2(\frac{1}{p_i})$$

Thus, $H(\text{X})$ and $H(\text{Y})$ are

$$H(\text{X}) = p(\text{X=0})log_2(\frac{1}{p(\text{X=0})}) + p(\text{X=1})log_2(\frac{1}{p(\text{X=1})}) = 0.59log_2(\frac{1}{0.59}) + 0.41log_2(\frac{1}{0.41}) \approx 0.977$$
$$H(\text{Y}) = p(\text{Y=0})log_2(\frac{1}{p(\text{Y=0})}) + p(\text{Y=1})log_2(\frac{1}{p(\text{Y=1})}) = 0.4log_2(\frac{1}{0.4}) + p0.6log_2(\frac{1}{0.6}) \approx 0.971$$

Below are the R codes for calculation.

```r
# Entropy

pX = c(cntX/n,1-cntX/n) # Probabilities of P(X=1) and P(X=0)
pY = c(cntY/n,1-cntY/n) # Probabilities of P(Y=1) and P(Y=0)

entropy = function(pX){ # Define a function to calculate entropy
  sum(pX*log(1/pX,2))
}
cat(' H(X) =', entropy(pX), '\n',
    'H(Y) =', entropy(pY))
```

```
##  H(X) = 0.9765005
##  H(Y) = 0.9709506
```

Question 3: correlations and covariance

Since X and Y are independent standard Gaussian random variables, and $U = X - Y$ and $V = 2X + 3Y$.It's easy to calculate

$$
\begin{aligned}
E[U] &= E[X] - E[Y] = 0 \\
E[V] &= 2E[X] + 3E[Y] = 0 \\
Cov(X,Y) &= E[XY] - E[X]E[Y] = 0
\end{aligned}
$$

And

$$
\begin{aligned}
V[U] &= Var(X) + Var(Y) - 2Cov(X,Y) = 2 \\
V[V] &= 4Var(X) + 9Var(Y) + 12Cov(X,Y) = 13
\end{aligned}
$$

Covariance

Futhermore, $Cov(X - Y, 2X + 3Y)$ can be calculated by using results above

$$
\begin{aligned}
Cov(U,V) &= Cov(X,2X) + Cov(X,3Y) + Cov(-Y,2X) + Cov(-Y,3Y) \\
&= 2Var(X) + Cov(X,Y) - 3Var(Y) \\
&= -1
\end{aligned}
$$

Below are the R codes for calculation.

```
nRuns <- 1000000 # Set runs

# Initialization
x <- rnorm(nRuns)
y <- rnorm(nRuns)
u <- x - y
v <- 2*x + 3*y

cat(cov(u, v)) #Covariance
```

```
## -1.002185
```

Correlation

Also, $Cor(U,V)$ can be calculated by using results above

$$
\begin{aligned}
Cor(U,V) &= \frac{Cov(X - Y, 2X + 3Y)}{\sqrt{Var(U)Var(V)}} \\
&\approx \frac{-1}{5.099} \\
&\approx -0.196
\end{aligned}
$$

Below are the R codes for calculation.

```r
cat(cor(u, v)) # Correlation
```

```
## -0.1966717
```

Question 4: maximum likelihood estimation of parameters

According to the question, the dataset comes from a Poisson distribution with unknown parameter λ . Thus, it is safely to figure out

$$p(\mathrm{k}|\lambda) = \frac{\lambda^k}{k!}e^(-\lambda)$$

And based on the theories of maximum likelihood estimation, the likelihood function is

$$\begin{aligned} L(\lambda) &= \prod_{i=1}^{n} p(\mathrm{k}|\lambda) \\ &= \prod_{i=1}^{n} \frac{\lambda^k}{k!}e^(-\lambda) \end{aligned}$$

And also the log likelihood function

$$\begin{aligned} l(\lambda) &= \sum_{i=1}^{n} ln(p(\mathrm{k}|\lambda)) \\ &= -\sum_{i=1}^{n} ln(k_i!) + \sum_{i=1}^{n} ln(\lambda) - n\lambda \end{aligned}$$

Then let $\frac{\partial}{\partial\lambda} ln(L) = 0$ and find the maximum value point, which told us the value of parameter λ

$$\begin{aligned} \lambda &= \frac{1}{n}\sum_{i=1}^{n} x_i \\ &= \frac{1}{10}\sum_{i=1}^{n} x_i \\ &= 3.9 \end{aligned}$$

Below are the R codes for simulation.

```
rawdata <- c(4,3,2,4,6,3,4,0,5,6,4,4,4,5,3,3,4,5,4,5) # Initialization

log_likelihood <- function(lambda, data){# Log likelihood function
  l = - sum(log(factorial(data))) + sum(data)*log(lambda) - length(data)*lambda
  return(-l)
}

#Use optimize to find out the parameter
rawdata_res <- optimize(log_likelihood, c(0, 100000), data = rawdata)
cat(rawdata_res$minimum)
```

```
## 3.899999
```

Question 5: central limit theorem

According to the question, the sequence is 10 i.i.d random variables from a Poisson distribution with $\lambda = 10$. For large λ, we have n ~ $Pois(\lambda)$ apprached n ~ $N(\lambda, \lambda)$ is this case,

$$p(\text{k|Pois(10)}) \approx p(\text{k|N(10,1)})$$

Consequently,

$$m_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

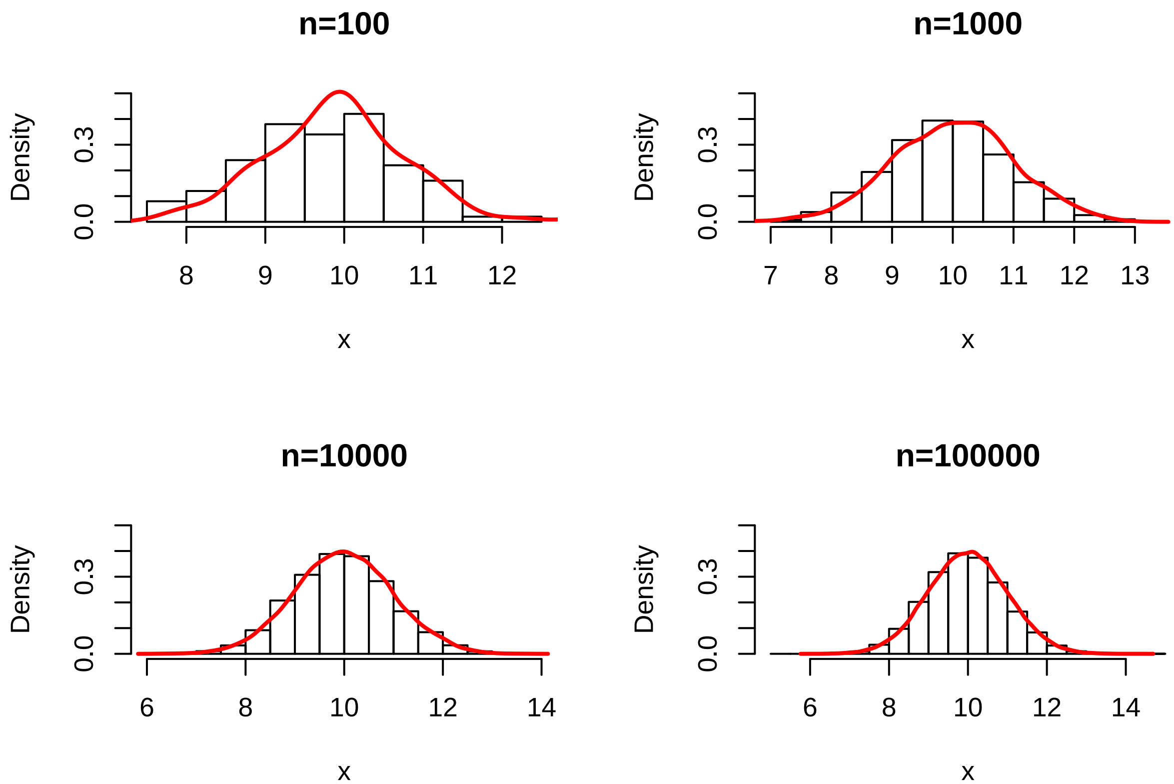
$$E[m_n] = E[\frac{1}{n}\sum_{i=1}^{n} X_i] = \frac{1}{n}\sum_{i=1}^{n} E[X_i] = E[X] = 10$$

$$V[m_n] = \frac{1}{n}V[X] = \frac{1}{10}10 = 1$$

Thus, the standard deviation is 1. Below are the R codes for simulation.

```r
par(mfrow=c(2,2))   # Initializaton for plot

# sample size 100
## Initialization
n1<-100
x_mean1 <- rep(NA,n1)

for (i in 1:n1){ # Loop to calculate means
  x_mean1[i]<-mean(rpois(10, 10))
}

hist(x_mean1, # Use histogram to show the probability of x
     main="n=100",
     probability=TRUE,
     xlab = 'x',
     ylim=c(0, 0.5))
x <- rnorm(n1, mean=10)
lines(density(x), col="red", lwd = 2) # Draw the normal distribution line


# sample size 1000
## Initialization
n2<-1000
x_mean2 <- rep(NA,n2)

for (i in 1:n2){ # Loop to calculate means
  x_mean2[i]<-mean(rpois(10, 10))
}

hist(x_mean2, # Use histogram to show the probability of x
     main="n=1000",
     probability=TRUE,
     xlab = 'x',
     ylim=c(0, 0.5))
x <- rnorm(n2, mean=10)
```

```r
lines(density(x), col="red", lwd = 2) # Draw the normal distribution line

# sample size 10000
## Initialization
n3<-10000
x_mean3 <- rep(NA,n3)

for (i in 1:n3){ # Loop to calculate means
  x_mean3[i]<-mean(rpois(10, 10))
}

hist(x_mean3, # Use histogram to show the probability of x
     main="n=10000",
     probability=TRUE,
     xlab = 'x',
     ylim=c(0, 0.5))
x <- rnorm(n3, mean=10)
lines(density(x), col="red", lwd = 2) # Draw the normal distribution line

# sample size 100000
## Initialization
n4<-100000
x_mean4 <- rep(NA,n4)

for (i in 1:n4){ # Loop to calculate means
  x_mean4[i]<-mean(rpois(10, 10))
}

hist(x_mean4, # Use histogram to show the probability of x
     main="n=100000",
     probability=TRUE,
     xlab = 'x',
     ylim=c(0, 0.5))
x <- rnorm(n4, mean=10)
lines(density(x), col="red", lwd = 2) # Draw the normal distribution line
```




```r
# Output
cat(' Mean of', n1 ,'samples :',mean(x_mean1), '\n',
    'Standard deviation of', n1 ,'samples :',sd(x_mean1), '\n',
    'Mean of', n2 ,'samples :',mean(x_mean2), '\n',
    'Standard deviation of', n2 ,'samples :',sd(x_mean2), '\n',
    'Mean of', n3 ,'samples :',mean(x_mean3), '\n',
    'Standard deviation of', n3 ,'samples :',sd(x_mean3), '\n',
    'Mean of', n4 ,'samples :',mean(x_mean4), '\n',
    'Standard deviation of', n4 ,'samples :',sd(x_mean4))
```

```
##  Mean of 100 samples : 9.797
##  Standard deviation of 100 samples : 0.9491575
##  Mean of 1000 samples : 9.9736
##  Standard deviation of 1000 samples : 0.9880178
##  Mean of 10000 samples : 10.00273
##  Standard deviation of 10000 samples : 1.002646
##  Mean of 1e+05 samples : 9.998535
##  Standard deviation of 1e+05 samples : 1.004548
```