



## 강원대학교 시험 답안지

		감독교수확인	이 두 호 	
2021학년도 1학기 기말고사	시험과목 : 데이터사이언스 프로그래밍	전공 : 컴퓨터공학과 학번: 201720970 성명: 권대환	출 제 자	이 두 호 
			채점성적	

1. (10점) 3차원 좌표공간의 두 점 A(2.7, 0.5, 1.2), B(3.3, 2.5, 0)에서 같은 거리에 있는 y축 위의 점의 좌표가 (0, a, 0)일 때, a의 값을 R 코딩으로 구하시오.

<실행 코드>

```
1 #exam 1
2 #Function Define
3 exam1 <- function(a)
4 {
5   sqrt((-2.7)^2 + (a - 0.5)^2 + (1.2)^2) - sqrt((3.3)^2 + (a - 2.5)^2)
6 }
7
8 #Find Value
9 result1 <- uniroot(exam1, lower = 2, upper = 3)$root
10 print(result1)
```

먼저 초기 RStudio에서 Stats Package가 이미 활성화되어 있으므로, uniroot 함수 사용에 문제가 없었다.

1번 문제에서는 점 A와 점 B 간의 거리를 구하는 식을 구하도록 하였으며, 결국 각 점의 거리를 뺀을 때 0이 되는 경우가 각 점 A와 B가 y축 위의 좌표 간의 거리가 같다고 볼 수 있으므로 3번 행에서는 제곱근을 사용해서 y축 좌표와 점 A의 거리와 점 B와의 거리의 차를 구했다.

이후 9번 행에서 uniroot 함수를 사용해서 3번 행에서 정의한 함수에 대한 해를 찾아내게 하였다.

범위는 2부터 3까지 찾도록 하였으며, \$root를 사용해서 실제 근만 result1 객체에 출력되도록 하였다.

마지막으로 10번 행에서 result1에 저장된 a의 값을 출력하게 된다.

<실행 결과>

```
> #Find Value
> result1 <- uniroot(exam1, lower = 2, upper = 3)$root
> print(result1)
[1] 2.040006
```

y축 위의 점의 좌표는 (0, 2.04, 0)이다.

2. (10점) 흰 공 3개, 검은 공 4개가 들어있는 주머니가 있다. 이 주머니에서 임의로 네 개의 공을 동시에 꺼낼 때, 흰 공 2개와 검은 공 2개가 나올 확률을 R 코딩으로 구하시오.

<실행 코드>

```
1 #exam 2
2 #combination Function Define
3 combination <- function(n, r)
4 {
5   factorial(n) / (factorial(r) * factorial(n-r))
6 }
7 #흰 공 3개, 검은 공 4개에서 각 공 2개를 임의 추출하는 확률이므로, 3C2 * 4C2 / 7C4가 된다.
8 result2 <- combination(3, 2) * combination(4, 2) / combination(7, 4)
9 print(result2)
```

먼저 추출에 대한 경우의 수를 계산하기 위해서, 코드의 3번 행에서 Combination에 대한 함수를 정의하였다.

다른 라이브러리에서 제공되는 Combination 함수를 사용하려 했지만, 사용 중인 R 언어 버전과 호환되지 않아 새로 정의하였다.

조합은 n개의 원소 중에서 r개의 원소를 뽑는 경우의 수를 말한다.

그리고 이 문제는 임의로 4개의 공을 임의 추출하는 것이므로 8번 행에서 문제의 답을 구한다.

8번 행의 처음 Combination 호출을 보면 (3, 2)로 칭해져 있으며, 이는 결국 3C2인 흰 공 3개 중 2개가 추출될 경우의 수를 나타내며,

그리고 Combination(4, 2) 즉, 4C2인 검은 공 4개 중 2개를 추출될 경우의 수와 곱함으로 사건이 일어날 경우의 수를 구했다.

이후 공 7개 중 4개를 추출할 전체 경우의 수를 나눠줌으로 임의 추출 시 흰 공, 검은 공 2개가 출력될 확률을 구할 수 있다.

<실행 결과>

```
> combination <- function(n, r)
+ {
+   factorial(n) / (factorial(r) * factorial(n-r))
+ }
> #흰 공 3개, 검은 공 4개에서 각 공 2개를 임의 추출하는 확률이므로, 3C2 * 4C2 / 7C4가 된다.
> result2 <- combination(3, 2) * combination(4, 2) / combination(7, 4)
> print(result2)
[1] 0.5142857
```

결과가 Result2에 저장되며, 9번 행 print 함수를 통해 결과가 출력된다. 답은 0.5142857이다.

3. (10점) 다음 정적분 값을 R 코딩으로 구하시오.

$$\int_e^{e^2} \frac{\ln x - 1}{x^2}$$

<실행 코드>

```
1 #exam 3
2 #문제의 정적분 함수 정의
3 formula1 <- function(x)
4 ~{
5   (log(x) - 1) / x^2
6 }
7 #Integrate 함수를 사용해서 e^2부터 e까지의 적분 실행
8 result3 <- integrate(formula1, lower = exp(1), upper = exp(2))$value
9 print(result3)
```

코드의 3번 행에서 계산하고자 하는 함수를 정의하였다.

이후 8번 행에서 적분 범위를  $e(\exp(1))$  부터  $e^2(\exp(2))$ 까지 정의하였으며, 3번 행에서 정의한 함수를 적분하도록 지정해주었으며, 적분 값만 출력시키기 위해서 integrate 함수 마지막에 \$value를 추가해주었다.

<실행 결과>

```
> formula1 <- function(x)
+ {
+   (log(x) - 1) / x^2
+ }
> #Integrate 함수를 사용해서 e^2부터 e까지의 적분 실행
> result3 <- integrate(formula1, lower = exp(1), upper = exp(2))$value
> print(result3)
[1] 0.09720887
```

integrate 함수를 통해 적분 값만 result3 객체에 저장되며, 저장된 데이터를 print 함수를 사용해서 출력하였다.

답은 0.09720887이다.

4. (10점) 다음 두 개 조건을 모두 만족시키는 음이 아닌 정수 a, b, c, d의 모든 순서쌍 (a, b, c, d)의 개수를 R 코딩으로 구하시오.

- I)  $a + b + c + d = 9$   
 II)  $d \leq 4$ 이고,  $c \geq d$ 이다.

<실행 코드>

```
1 #exam 4
2 #combination Function Define
3 combination <- function(n, r)
4 {
5   factorial(n) / (factorial(r)*factorial(n-r))
6 }
7 #a + b + c' = 9 - 2d에 따라, d = 0, d = 1, d = 2, d = 3, d = 4의 확률을 구한다.
8 #c' <= 0
9 result4 <- 0
10 for(i in seq(1, 9, 2))
11 {
12   result4 <- result4 + print(combination(i + 2, i))
13 }
14 print(result4)
```

문제에서 제시한 조건 1과 조건 2에 부합하는 순서쌍은  $a + b + c' = 9 - 2d$ 와 같다고 볼 수 있다.

그리고 c는 d보다 크거나 같아야 한다는 조건을 지켜줘야 한다.

모든 순서쌍이 중복할 수 있으면서, 제시한 조건만 맞으면 되므로, 먼저 d가 0일 경우,  $a + b + c' = 9$ 이므로 중복조합인 3H9에 해당한다.

구체적으로 d가 0이라면, 조건에 따라 서로 다른 숫자 9개에서 3개를 택하는 것과 같으므로  $3H9 == 11C9$ 으로 계산한다.

다음으로 d가 1이라면,  $a + b + c' = 7$ , d가 2라면  $a + b + c' = 5$ , d가 3라면  $a + b + c' = 3$ , d가 4라면  $a + b + c' = 1$ 로 표현 할 수 있다.

이는 각  $3H9 == 11C9$ ,  $3H7 == 9C7$ ,  $3H5 == 7C5$ ,  $3H3 == 5C3$ ,  $3H1 = 3C1$ 의 중복조합을 나타낸다.

이후 반복문을 거치면서 기존에 만들어둔 Combination 함수를 통해 d = 0:4까지 계산 후 이를 더해서 순서쌍의 개수를 출력한다.

<실행 결과>

```
> result4 <- 0
> for(i in seq(1, 9, 2))
+ {
+   result4 <- result4 + print(combination(i + 2, i))
+ }
[1] 3
[1] 10
[1] 21
[1] 36
[1] 55
> print(result4)
[1] 125
```

순서쌍의 개수는 125개다.

5. (10점) 한 개의 동전을 7번 던질 때, 다음 두 개의 조건을 모두 만족시킬 확률을 R 코딩으로 구하시오.

I) 앞면이 3번 이상 나온다.

II) 앞면이 연속해서 나오는 경우가 있다.

#### <실행 코드>

```

1  #exam 5
2  #combination Function Define
3  combination <- function(n, r)
4  {
5    factorial(n) / (factorial(r)*factorial(n-r))
6  }
7  #모든 경우의 수를 더하고, 연속되지 않는 경우를 뺀으로 확률을 구하였다.
8  coin <- function(n)
9  {
10   calc_result <- 0
11   calc_result <- calc_result + combination(7, n) #7C3, 7C4, 7C5, 7C6, 7C7
12   if(n == 3) #4H2
13   {
14     calc_result <- calc_result - combination(5, n) #앞면이 3번 나왔을 경우의 불연속 앞면 배치 경우의 수, 5C2 == 5C3
15   }
16   else if(n == 4)
17   {
18     calc_result <- calc_result - combination(4, 4) #앞면이 4번 나왔을 경우의 F,B,F,B,F,B,F (불연속 앞면 배치)
19   }
20   print(calc_result)
21 }
22
23 result5 <- 0
24 for(i in 3:7)
25 {
26   result5 <- result5 + coin(i)
27 }
28 result5 <- result5 / 2^7
29 print(result5)

```

경우의 수를 계산하기 위해서 코드의 3번 행에서 Combination 함수를 정의하였다.

코드 8번 행부터 문제에서 제시한 조건에 대한 확률을 계산하는 함수를 정의하였으며, 10번 행에서 계산된 경우의 수를 모두 저장해서 함수 밖으로 출력하기 위해 선언한 객체다.

이후 11번 행에서 앞면이 3번 이상 나올 확률을 calc\_result에 저장하게 된다.

그리고 7, 6, 5번 앞면이 나올 때는 연속되어 나오지만 반면에 3번, 4번 앞면이 나온다면 연속되지 않는 경우의 수가 존재하므로 이에 대한 불연속 경우를 빼줘야 한다.

해당 함수의 12번 행에서 호출된 인자 값이 3이라면, 같은 말로 3번 앞면이 나오는 경우를 생각해보면, 결국 X, 뒤, X, 뒤, X, 뒤, X, 뒤, X 와 같이 X의 위치에 앞면이 나올 수 있으므로, 5C3(combination(5, 3))을 곱셈에서 빼주게 된다.

그리고 앞면이 4번 나올 경우의 수에서는 앞, 뒤, 앞, 뒤, 앞, 뒤, 앞의 경우만 존재하므로, 1개의 경우의 수를 뺀으로써 값을 구하게 된다.

마지막으로 23번 행에서는 출력된 함수 데이터를 저장하기 위한 객체를 선언하였으며, 24번 행에서는 반복문을 사용해서 앞면이 3번 나올 확률부터 7번 나올 확률까지 함수를 호출해서 23번 행의 객체에 모두 저장한다.

이후 28번 행에서 전체 사건 수인  $2^7$ 을 나눠줌으로 29번 행에서 계산한 확률을 출력한다.

#### <실행 결과>

```

> result5 <- 0
> for(i in 3:7)
+ {
+   result5 <- result5 + coin(i)
+ }
[1] 25
[1] 34
[1] 21
[1] 7
[1] 1
> result5 <- result5 / 2^7
> print(result5)
[1] 0.6875

```

result5를 0으로 선언하였으며, coin 함수의 print 함수 때문에 각 경우의 수가 모두 출력되며, 이후 전체 사건인  $2^7$ 을 나눠줌으로 답을 구한다.

동전을 7번 던졌을 때 그리고 문제에서 제시한 두 가지 조건이 나올 확률은 0.6875이다.

6. (10점) 확률변수  $X$ 는 정규분포  $N(10, 2^2)$ , 확률변수  $Y$ 는 정규분포  $N(m, 2^2)$ 을 따르고, 확률변수  $X$ 와  $Y$ 의 확률밀도함수는 각각  $f(x)$ 와  $g(x)$ 이다.  $f(12) \leq g(20)$ 를 만족시키는  $m$ 에 대하여  $\Pr\{21 \leq Y \leq 24\}$ 의 최댓값을 R 코딩으로 구하시오.

<실행 코드>

```
1 #exam 6
2 #정규 분포 공식  $x, m = \text{mean}, o = \text{표준편차}$ 
3 pro_distribute <- function(x, m, o)
4 {
5    $(2 * \text{sqrt}(2 * \pi))^{-1} * \exp(-(x-m)^2 / (2 * (o^2)))$ 
6 }
7
8 #계산 후의 정규분포 식 이를 기반으로 답을 구한다.
9 result_distribute <- function(x)
10 {
11    $(2 * \text{sqrt}(2 * \pi))^{-1} * \exp(-(x-22)^2 / (2 * (2^2)))$ 
12 }
13
14 calc6 <- function(y, z, y_prob1, y_prob2) #x <= y, z == 표준편차
15 {
16   #예상되는 m의 범위
17   y_data1 <- y - z
18   y_data2 <- y + z
19
20   #f(12) <= g(20)이므로 f(12)와 g(20)의 그래프를 그려서 범위 내 최대 값을 구한다.
21   curve(pro_distribute(x, 10, 2), xlim = c(5, 30), lty = 1, type = "l")
22   curve(pro_distribute(x, 12, 2), xlim = c(5, 30), col = "green", lty = 1, type = "l", add = TRUE)
23
24   curve(pro_distribute(x, 20, 2), lty = 1, type = "l", add = TRUE)
25
26   abline(v = y_prob1)
27   abline(v = y_prob2) #maximum m <- (y_prob1 + y_prob2) / 2 == 22.5
28   curve(pro_distribute(x, 22, 2), col = "red", lty = 1, type = "l", add = TRUE)
29
30   #curve(result_distribute, xlim = c(15, 30), col = "red", lty = 1, type = "l")
31
32   #abline(v = y_data2 - z, col = "red")
33   #abline(v = y_data2 + z, col = "red") #m range check
34
35   snd1 <- (y_prob1 - y_data2) / z
36   snd2 <- (y_prob2 - y_data2) / z
37
38   m_range <- y_data1
39   m_range %<>% append(y_data2)
40   print(m_range)
41   snd_result <- snd1
42   snd_result %<>% append(snd2)
43
44   print(snd_result)
45 }
46
47 calc6(20, 2, 21, 24)
48
49 #최대 값 출력
50 result6 <- integrate(result_distribute, lower = 21, upper = 24)$value
51 print(result6)
```

코드의 3번 행에서 인자값을 받는 정규분포 공식을 함수로 정의하였으며, 인자의  $m$ 은 평균,  $o$ 는 표준 편차를 말한다.

9번 행에서는 계산된 평균과 범위를 기반으로 정규분포를 적분할 때 사용하는 함수이다. (이미 계산된 평균, 표준편차가 들어있다)

14번 행에는 문제 6번을 풀이하기 위한 함수를 새로 정의하였다.

이후 17번과 18번 행에서 확률변수  $Y$ 에 대한 평균 가능 범위를 나타내는데, 물론 이는  $f(12) \leq g(20)$  조건에 따라  $g(20)$ 이 평균에서 표준 편차만큼 떨어진 곳의 값보다 많이 크려면  $18 \leq m \leq 22$ 와 같은 범위를 가져야 한다는 것이다.

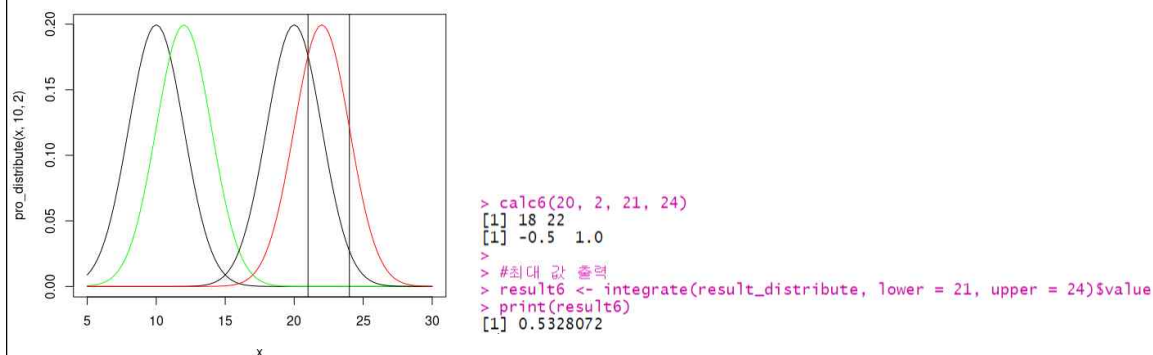
그리고  $\Pr\{21 \leq Y \leq 24\}$ 의 최댓값을 구하라고 하였으므로 본래는  $(21 + 24) / 2 = 22.5$ 의 계산을 통해 최댓값의 위치를 파악할 수도 있지만,

21번 행부터 28번 행까지  $f(10)$ ,  $f(12)$ ,  $g(20)$ ,  $g(22)$ 의 그래프를 그렸으며, 21과 24의 범위 내에서 최댓값을 확인하도록 작성하였다.

35번 36번 행은 표준정규분포의 범위로 변환해주는 것이다. 표준정규분포표가 있다면 해당 수치를 보고 문제를 풀 수도 있다.

마지막으로 38번부터 44번까지는  $m$ 의 범위와 표준정규분포 범위의 값을 출력해주며 50번 행에서는 평균 22에서의 21, 24 범위를 적분한 값을 저장함으로 51번 행에서 최댓값을 출력하게 된다.

<실행 결과>



실제 그래프에서 21과 24 범위에서 최댓값을 가지는  $m$  값은 22.5임을 알 수 있다.

그러나  $18 \leq m \leq 22$ 의 범위를 가지고 있으므로,  $m$ 은 22에서 최댓값을 가진다고 볼 수 있다.

그리고 해당  $m$ 이 22인 분포에서 21과 24 범위를 적분했을 때, 정답은 0.5328072이다.

7. (20점) 아래 유튜브 영상을 시청하고 수학 함수  $f$ 를 입력하면 함수  $f$ 의 그래프를 그려주는 R 함수 `my.graph()`를 구현하고, `my.graph()`를 이용하여

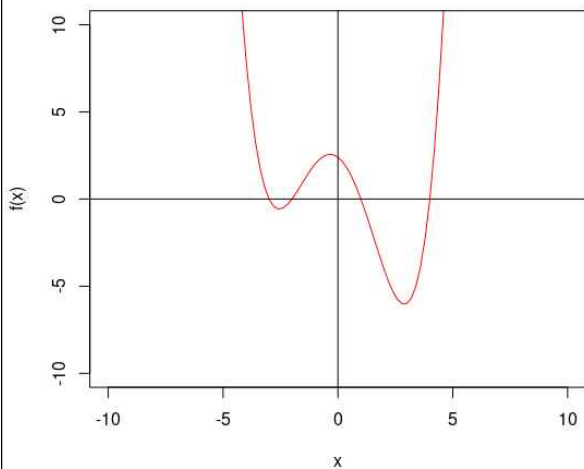
$f(x) = \frac{x^4 - 15x^2 - 10x + 24}{10}$ 를 그리시오.

#### <실행 코드>

```
1 #exam 7
2 my.graph <- function(x, n)
3 {
4   plot(x, xlim=c(-n,n), ylim=c(-n,n), xlab="x", ylab="f(x)", lty = 1, type = 'l', col = 'red')
5   abline(v=0)
6   abline(h=0)
7 }
8 f <- function(x)
9 {
10  (x^4-15*x^2-10*x+24)/10
11 }
12 my.graph(f, 10)
```

코드의 2번 행에서 `my.graph` 함수를 정의하였으며, 4번 행에서 그래프 출력함수인 `plot`을 사용해서  $x$  인자로 입력된 함수의 그래프를 그리도록 하였다. 추가로 문제에서 요구한 입력값에 따른 그래프 출력 범위 정의를 위해 입력된  $n$  인자의 숫자 데이터값을 사용해서 그래프의  $x$ ,  $y$  범위를 재정의하였으며, `lty`를 1(실선)로, 그리고 `type`을 l(Line)으로 설정하였으며, 추가로 `col(color)`을 `red(빨간색)`로 정의해둠으로 빨간 선의 그래프를 그리게 되었다. 마지막으로 5, 6번 행에서 각 수직, 수평의 0 좌표를 기준으로 선을 그리도록 하였다. 문제에서는 구체적으로  $x$ ,  $y$  범위에 대한 구체적인 정의를 하지 않았으므로 해당 코드에서는  $x$ ,  $y$  모두  $(-10, 10)$ 의 범위를 가지게 정의하였다.

#### <실행 결과>



각 사분면이 좌표 0을 기준으로 명확하게 나누어져 있으며, 해당 함수를 빨간 선을 출력하도록 하였으며, 요구사항에 맞게 잘 출력된 것을 확인할 수 있다.

8. (20점) 2019년 3월부터 5월까지 치러진 한화이글스의 경기에 대해서 아래의 5개 항목을 조사하고, eagles라는 이름으로 데이터 프레임을 만드시오. 단, 취소 경기는 제외한다.

조사항목 : 한화이글스의 안타 수, 한화이글스의 실책 수, 상대 팀의 안타 수, 상대 팀의 실책 수, 한화이글스의 경기 결과(승 또는 패)

조사한 결과를 토대로 한화이글스의 경기 결과(승 또는 패)를 예측하는 인공신경망 모델을 만들고, 이 인공신경망 모델을 바탕으로 아래의 경우에 대해 한화이글스의 경기 결과를 예측하시오.

한화 이글스 안타 수 : 14개

한화 이글스 실책 수 : 17개

상대 팀 안타 수 : 7개

상대 팀 실책 수 : 5개

#### <실행 코드>

```
1 #exam 8
2 load("~/eagles.RData")
3
4 #행 개수 기반 임의 추출 -> sample_num에 저장
5 sample_num <- sample(1:nrow(eagles), nrow(eagles)*0.9)
6
7 #데이터 검증을 위한 Train, Test 데이터를 구분하였음
8 eagles.train <- eagles[sample_num, ]
9 eagles.test <- eagles[-sample_num, ]
10
11 #nnet 함수를 사용하여, input, output에 상응되는 데이터 학습
12 nnet_object <- nnet(result ~., data=eagles.train, size=20, maxit=10000)
13
14 #Test 데이터 기반으로 승부 예측을 해보았음
15 prepare.prediction <- predict(nnet_object, eagles.test[, -5], type = "class")
16
17 #상응되는 경기 결과들이 출력됨
18 print(prepare.prediction)
19
20 #해당 문제의 예측 경기 지표를 정의하였음
21 real.data <- data.frame(14, 1, 7, 5)
22 colnames(real.data) <- c("eagles.hits", "eagles.booted", "rival.hits", "rival.booted")
23
24 #방금 정의한 데이터를 기반으로 경기 승/패를 예측함
25 result8 <- predict(nnet_object, real.data, type = "class")
26 print(result8)
```

코드의 행 2번에서 사전에 저장해둔 Eagles 객체를 파일로부터 로드한다.

이후 로드한 객체의 행 번호 기반으로 sample 함수를 사용해서 sample\_num에 임의의 행 번호를 행 개수의 90%만큼 저장한다.

그리고 학습 데이터와 테스트 데이터를 구분하기 위해서 sample\_num에 존재하는 행 번호를 학습 데이터 객체인 eagles.train에 저장하도록 하였으며, 9번 행에서는 eagles.test 객체에 sample\_num에 존재하지 않는 행 번호를 저장함으로 추후 학습 데이터 검증에 사용된다.

12번 행에서 nnet 함수를 사용해서 각 Input(eagles.hits, eagles.booted, rival.hits, rival.booted)에 맞는 Output(result) 데이터를 학습한다.

15번 행에서 학습된 Weight 기반으로 학습하지 않은 eagles.test 데이터를 기반으로 경기 결과를 예측해본다.

18번 행에서 학습 결과를 확인함으로, 실제 경기 결과와 비교할 수 있도록 하였다.

여기서 행 번호가 임의 순서로 들어가 있으므로 원본 객체의 행 번호와 비교했을 때, 학습이 제대로 되었는지 검증할 수 있었다.

마지막으로 문제에서 제시한 팀 성적에 따른 경기 예측을 위해서, 21번 행에서 데이터 정의, 22번 행에서 필드를 정의해주었다.

25번 행에서 만들었던 모델에 실제 경기 예측을 진행하며, 26번 행에서 저장된 예측 결과를 출력한다.

#### <실행 결과>

```
> #Test 데이터 기반으로 승부 예측을 해보았음
> prepare.prediction <- predict(nnet_object, eagles.test[, -5], type = "class")
>
> #상응되는 경기 결과들이 출력됨
> print(prepare.prediction)
[1] "win" "win" "loss" "win" "win" "loss" "win"
>
> #해당 문제의 예측 경기 지표를 정의하였음
> real.data <- data.frame(14, 1, 7, 5)
> colnames(real.data) <- c("eagles.hits", "eagles.booted", "rival.hits", "rival.booted")
>
> #방금 정의한 데이터를 기반으로 경기 승/패를 예측함
> result8 <- predict(nnet_object, real.data, type = "class")
> print(result8)
[1] "win"
```

처음은 테스트 데이터에 대한 경기 예측 결과를 출력하며, 두 번째 출력은 문제에서 제시한 조건에 따른 경기 예측 결과가 출력된다.

정답은 승리이다.

그리고 여러 번 학습시켰을 때 테스트 데이터 추출로 인한 학습 데이터가 변경됨으로 학습 결과가 달라지는 문제가 존재하며,

이는 테스트 데이터 추출을 하지 않고, 실제 학습 모델을 만들었을 때 경기 예측 결과가 달라지는 문제가 없었다.