Rfinal

김어진(201917959)

2021 12 21

1번 문제 < 회귀분석 >

(1) 벡터생성

```
e=rnorm(1000)
head(e)
```

```
## [1]  1.70193335 -0.87185138  0.09513676 -1.13209800  0.02938743 -0.59294394
```

(2) 벡터생성

```
t=c()
X_1=c()
X_2=c()
for(i in 1:1000){
    t[i]=2*pi*i/1000
    X_1[i]=sin(t[i])
    X_2[i]=cos(4*t[i])
}

head(X_1)
```

```
## [1] 0.006283144 0.012566040 0.018848440 0.025130095 0.031410759 0.037690183
```

```
head(X_2)
```

```
## [1] 0.9996842 0.9987370 0.9971589 0.9949510 0.9921147 0.9886517
```

(3) y_i를 계산하고 (t_i, y_i)를 시각화

- y_i 계산

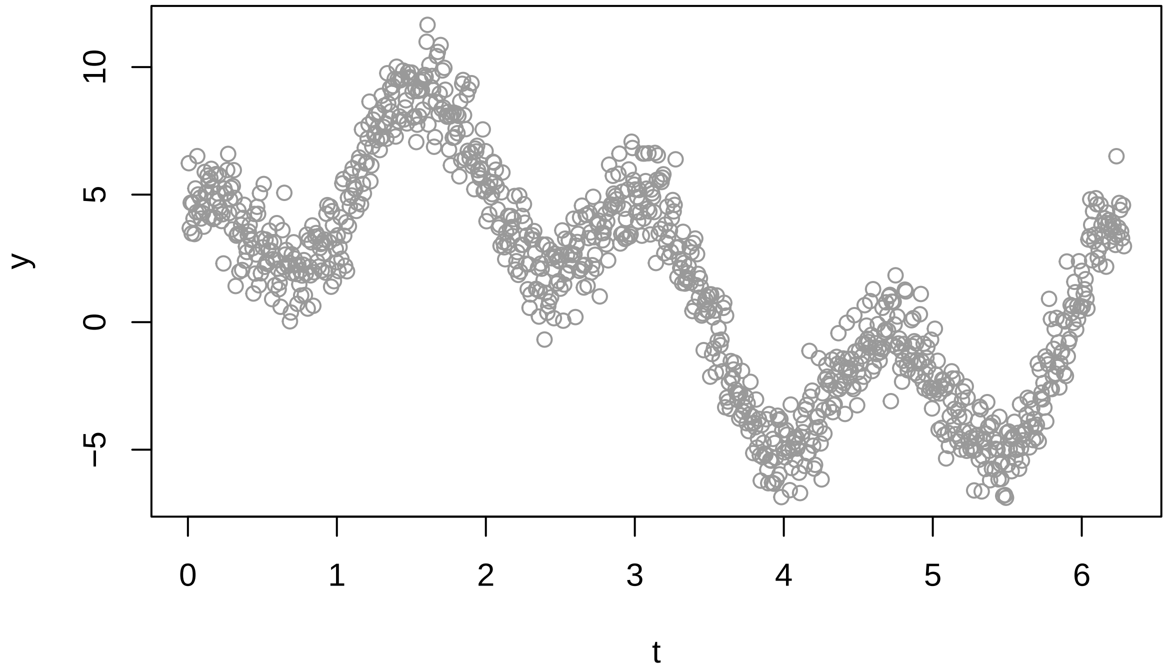
```
t=c()
X_1=c()
X_2=c()
y=c()
for(i in 1:1000){
    t[i]=2*pi*i/1000
    X_1[i]=sin(t[i])
    X_2[i]=cos(4*t[i])
    y[i]=1.5 + 5*X_1[i] + 3*X_2[i] + e[i]
}

head(y)
```

```
## [1] 6.232402 3.687190 4.680856 3.478406 4.662785 4.061462
```

- (t_i, y_i) 시각화

```
plot(t,y,col='gray60')
```



(4) 매트릭스 생성

```
X=cbind(1, X_1, X_2)
head(X)
```

```
##                    X_1       X_2
## [1,] 1 0.006283144 0.9996842
## [2,] 1 0.012566040 0.9987370
## [3,] 1 0.018848440 0.9971589
## [4,] 1 0.025130095 0.9949510
## [5,] 1 0.031410759 0.9921147
## [6,] 1 0.037690183 0.9886517
```

(5) 매트릭스를 만들고 $\mathbf{X}\boldsymbol{\beta}$를 계산, 벡터화하고 시각화

- $\beta = \begin{bmatrix} 1.5 \\ 5 \\ 3 \end{bmatrix}$

```
beta=rbind(1.5,5,3)
beta
```

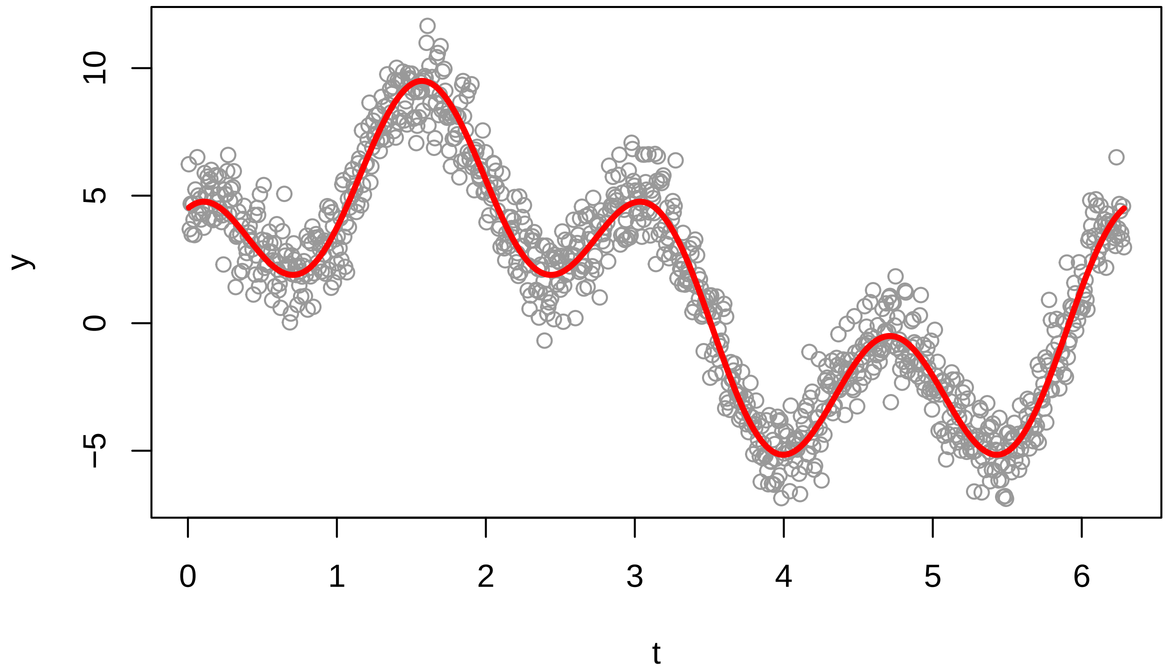
```
##      [,1]
## [1,]  1.5
## [2,]  5.0
## [3,]  3.0
```

- $\mathbf{X}\boldsymbol{\beta}$

```
X %*% beta ->Xbeta
head(Xbeta)
```

```
##          [,1]
## [1,] 4.530468
## [2,] 4.559041
## [3,] 4.585719
## [4,] 4.610504
## [5,] 4.633398
## [6,] 4.654406
```

```
plot(t,y,col='gray60')
lines(t,Xbeta, col='red',lwd=3)
```



(6) $\hat{\beta}$를 계산하고 β와 비교

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
dim(y)= c(1000,1)

X_T=t(X)
X_T %*% X %>% solve() %*% X_T %*% y ->betahat
betahat
```

```
##         [,1]
##     1.542234
## X_1 4.935844
## X_2 2.999107
```

$$\hat{\beta} = \begin{bmatrix} 1.495225 \\ 5.039034 \\ 2.925439 \end{bmatrix}$$

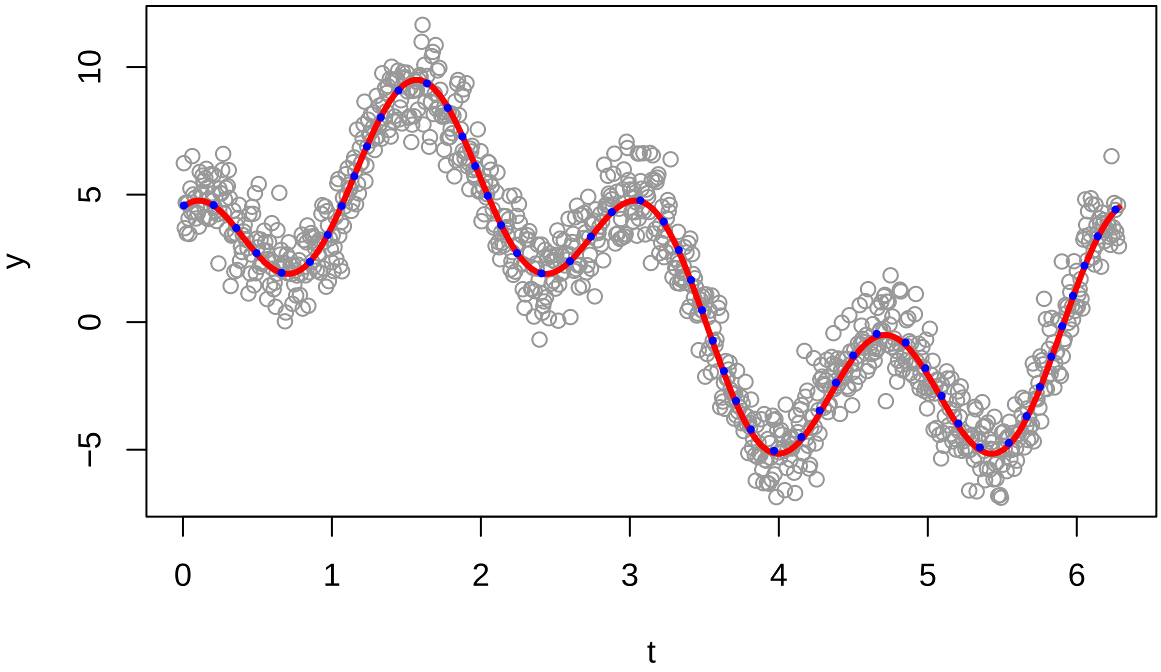
$$\beta = \begin{bmatrix} 1.5 \\ 5 \\ 3 \end{bmatrix}$$

약간은 차이가 나지만 비슷하다.

(7) $\mathbf{X}\hat{\beta}$를 계산

```
X %*% betahat ->Xbetahat
Xbetahat=as.vector(Xbetahat)

plot(t,y,col='gray60')
lines(t,Xbeta, col='red',lwd=3)
lines(t,Xbetahat,lty=3, col='blue', lwd=4)
```

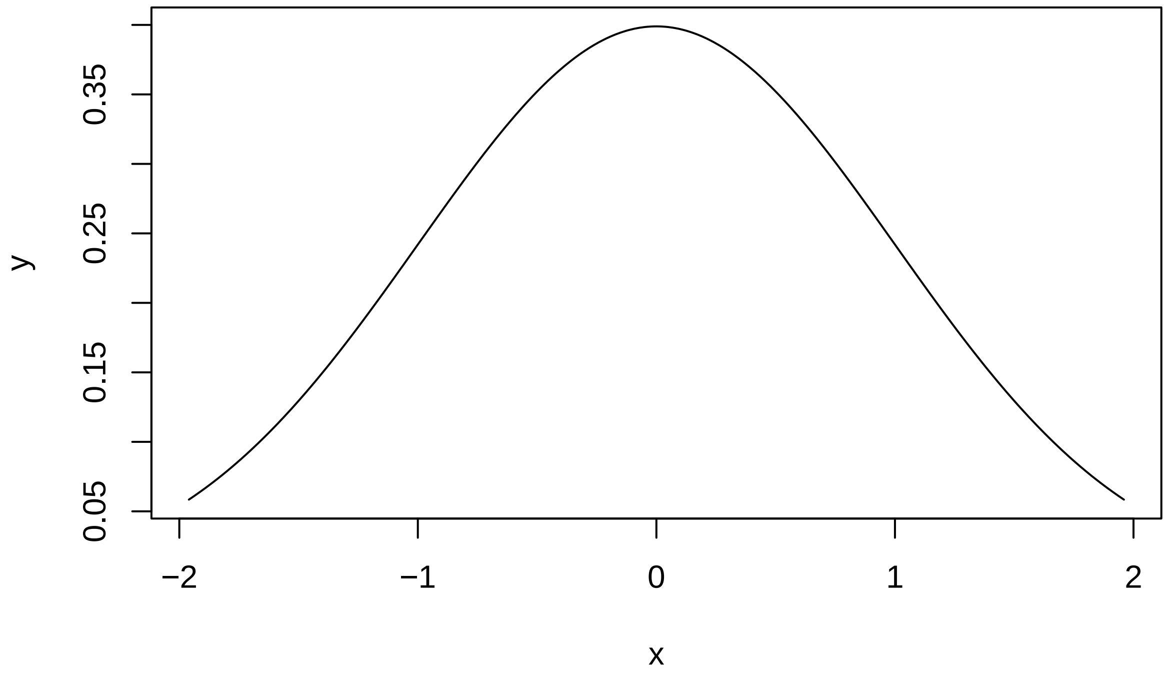

y
10
5
0
−5
0
1
2
3
4
5
6
t

2번 문제 < 몬테카를로 적분>

(1) runif를 이용해 몬테카를로 적분 계산

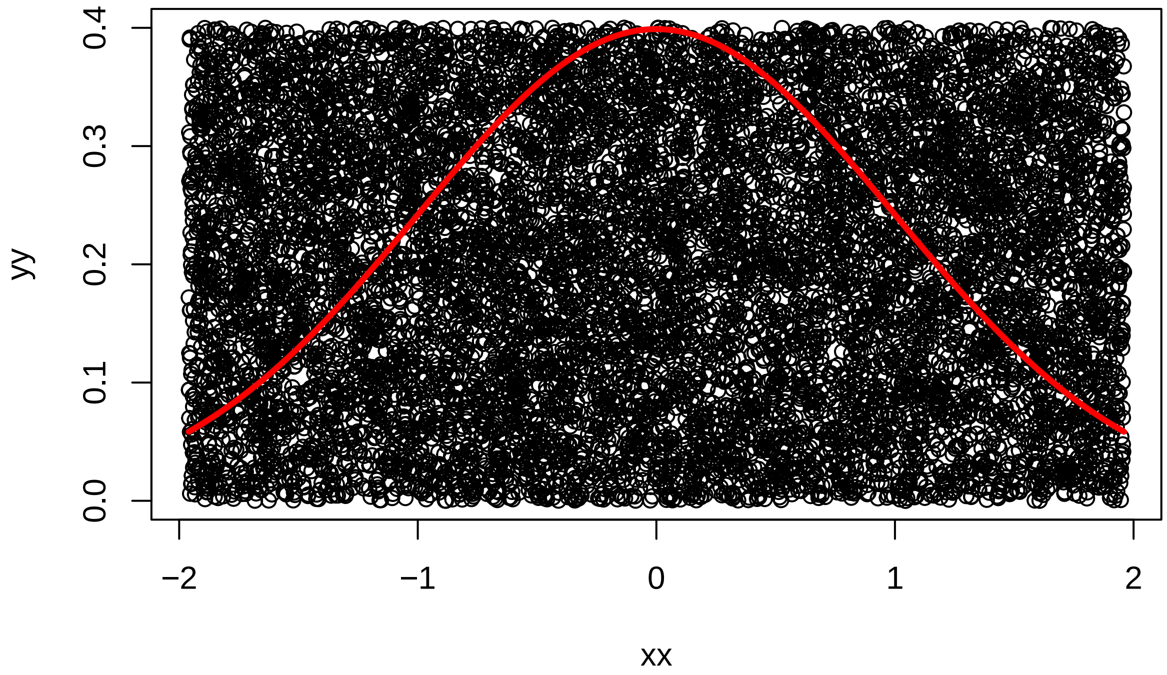
```
x=seq(from=-1.96, to=1.96, by=0.01)
y=1/sqrt(2*pi) *exp(-1/2 *x^2)

plot(x,y,type='l')
```



```
xx=runif(10000, min=-1.96, max=1.96)
yy=runif(10000, min=0, max=0.40)

plot(xx,yy)
lines(x,y, col='red', lwd=3)
```



```
test = function(xx,yy){
    yy < 1/sqrt(2*pi) *exp(-1/2 *xx^2)
}

tst=c()
for(i in 1:10000) tst[i]= test(xx[i],yy[i])
head(tst)
```

```
## [1] TRUE TRUE TRUE TRUE TRUE TRUE
```

```
sum(tst)
```

```
## [1] 6051
```

전체 10000개 중에서 6055개가 그래프 아래에 위치하므로 $\frac{s}{4*0.4} \approx \frac{sum(tst)}{10000}$ 구하는 넓이는 대략적으로

```
s=sum(tst)/10000*4*0.4
s
```

```
## [1] 0.96816
```

(2) rnorm()을 이용해 확률변수를 count

```
a=rnorm(1000)
a= as_tibble(a)
a %>% filter(-1.96 <value, value<1.96) %>% count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1   951
```

확률변수 1000개 중 945개가 구간(-1.96, 1.96) 사이에 위치한다.

3번 문제 < 징검다리 >

- TYPE A: 10,9,8,. . .,3,2,1 순

```
### 변수들의 모음
ARR = c('N1','N2','N3','N4','N5','N6','N7','N8','A','N10')#10,A,8,순
SURV = 10
PLAYER = ARR[SURV]
STAGE = 0
PROB = 0.95
TOSSRSLT = NA

### 함수들의 모음
toss = function(p) rbinom(n=1,size=1,prob=p) %>% as.logical
reset = function(){
    TOSSRSLT <<- NA
    SURV <<- 10
    STAGE <<- 0
    PLAYER <<- ARR[SURV]
}
record = function(){
    list(PRE_TOSSRSLT=TOSSRSLT, SURV=SURV, STAGE=STAGE, PLAYER=PLAYER)
}
go = function(){
    PROB <<- 0.5+ (PLAYER=='A')*0.45
    TOSSRSLT <<- toss(PROB)
    if (TOSSRSLT==FALSE) SURV <<- SURV - 1
    STAGE <<- STAGE + 1
    PLAYER <<- ARR[SURV]
    if(SURV==0) reset()
}
gogo = function() for(i in 1:20) go()

gogo_history = function(){
    rslt_ = as_tibble(record())
    for(i in 1:20){
        go()
        rslt_ = rbind(rslt_, as_tibble(record()))
    }
    print(rslt_)
}

simulate_once = function(){
    reset()
    gogo()
    return(record()$SURV )
}


### body
simrslt = c()
for (i in 1:100000) simrslt[i] = simulate_once()
mean(simrslt)
```

```
## [1] 6.80226
```

- TYPE A에서 8번 참가자는 생존자가 8명이상일 때 생존가능하다.

- TYPE B: 1,2,3,. . .,8,9,10순

```
ARR = c('N10','A','N8','N7','N6','N5','N4','N3','N2','N1')#1,2,3순
SURV = 10
PLAYER = ARR[SURV]
STAGE = 0
PROB = 0.95
TOSSRSLT = NA

### 함수들의 모음
toss = function(p) rbinom(n=1,size=1,prob=p) %>% as.logical
reset = function(){
    TOSSRSLT <<- NA
    SURV <<- 10
    STAGE <<- 0
    PLAYER <<- ARR[SURV]
}
record = function(){
    list(PRE_TOSSRSLT=TOSSRSLT, SURV=SURV, STAGE=STAGE, PLAYER=PLAYER)
}
go = function(){
    PROB <<- 0.5+ (PLAYER=='A')*0.45
    TOSSRSLT <<- toss(PROB)
    if (TOSSRSLT==FALSE) SURV <<- SURV - 1
    STAGE <<- STAGE + 1
    PLAYER <<- ARR[SURV]
    if(SURV==0) reset()
}
gogo = function() for(i in 1:20) go()

gogo_history = function(){
    rslt_ = as_tibble(record())
    for(i in 1:20){
        go()
        rslt_ = rbind(rslt_, as_tibble(record()))
    }
    print(rslt_)
}

simulate_once = function(){
    reset()
    gogo()
    return(record()$SURV )
}


### body
simrslt = c()
for (i in 1:100000) simrslt[i] = simulate_once()
mean(simrslt)
```

```
## [1] 3.0517
```

- TYPE B에서는 8번 참가자는 생존자가 3명 이상일 때 생존가능하다.
- 따라서 8번 참가자는 TYPE B에서 살아남을 확률이 높다.

4번 문제 < COVID19 >

```
df=read_csv('https://raw.githubusercontent.com/guebin/2021IR/master/_notebooks/covid19.c
```

```
## Rows: 12294 Columns: 5
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): prov
## dbl (4): year, month, day, cases
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
head(df)
```

```
## # A tibble: 6 x 5
##    year month   day prov  cases
##   <dbl> <dbl> <dbl> <chr> <dbl>
## 1  2020     1    20 서울      0
## 2  2020     1    20 부산      0
## 3  2020     1    20 대구      0
## 4  2020     1    20 인천      1
## 5  2020     1    20 광주      0
## 6  2020     1    20 대전      0
```

(1) 2020년 확진자 총합과 2021년 확진자 총합은?

```
df %>% group_by(year) %>% summarise(ttl_case=sum(cases))
```

```
## # A tibble: 2 x 2
##    year ttl_case
##   <dbl>    <dbl>
## 1  2020    60726
## 2  2021   396886
```

2020년에는 60726명, 2021년에는 396886명

(2) 20년 2월 1일~15일까지의 기간동안 지역별 확진자의 총합과 가장 많은 확진자가 발견된 지역은?

```
province=c('서울','부산','대구','인천','광주','대전','울산','세종','경기',
           '강원','충북','충남','전북','전남', '경북','경남','제주','검역')

tb=tibble(province=province, ttlcases=c(rep(0,length(province))))
a=c()
df %>% filter(year==2020 & month==2 & (day %in% c(1:15))) -> day1to15
for(i in 1:length(province)){
    day1to15 %>% filter(prov==province[i]) %>% summarise(ttl=sum(cases))-> a[i]
    tb[i,2]= a[i]
}
tb
```

```
## # A tibble: 18 x 2
##    province ttlcases
##    <chr>       <dbl>
##  1 서울            5
##  2 부산            0
##  3 대구            0
##  4 인천            0
##  5 광주            2
##  6 대전            0
##  7 울산            0
##  8 세종            0
##  9 경기            9
## 10 강원            0
## 11 충북            0
## 12 충남            0
## 13 전북            0
## 14 전남            1
## 15 경북            0
## 16 경남            0
## 17 제주            0
## 18 검역            0
```

```
max(tb$ttlcases)
```

```
## [1] 9
```

2020년 2월 1일~15일동안 가장 많은 확진자가 발견된 지역은 경기(9)이다.

(3) 2020년 2월 16일~29일까지의 기간동안 지역별 확진자의 총합과 가장 많은 확진자가 발견된 지역은?

```
df %>% filter(year==2020 & month==2 & (day %in% c(16:29))) -> day16to29
for(i in 1:length(province)){
    day16to29 %>% filter(prov==province[i]) %>% summarise(ttl=sum(cases))-> a[i]
    tb[i,2]= a[i]
}
tb
```

```
## # A tibble: 18 x 2
##    province ttlcases
##    <chr>       <dbl>
##  1 서울           62
##  2 부산          75
##  3 대구        2055
##  4 인천           5
##  5 광주           7
##  6 대전          13
##  7 울산          17
##  8 세종           1
##  9 경기          65
## 10 강원           7
## 11 충북          10
## 12 충남          48
## 13 전북           4
## 14 전남           1
## 15 경북         472
## 16 경남          59
## 17 제주           2
## 18 검역           0
```

```
max(tb$ttlcases)
```

```
## [1] 2055
```

2020년 2월 16일~29일동안 가장 많은 확진자가 발견된 지역은 대구(2055)이다.