Example1(sh)

2019712088 이승환

2020 10 25

```r
# Statistical Data Mining Example 1 ==================================
setwd("C:/Users/ghkstbd/Desktop/2020 고급데이터마이닝/R Example 1")
############### Q1. #################
# red points form training daaset
redpoints=read.table("trainred.txt",sep="\t",header=FALSE)

# green points from training dataset
greenpoints=read.table("traingreen.txt",sep="\t",header=FALSE)

# red points from test dataset
redtestpoints=read.table("testred.txt",sep="\t",header=FALSE)

# green points from test dataset
greentestpoints=read.table("testgreen.txt",sep="\t",header=FALSE)

# Combine red and green points.
dim(redpoints) # 100 by 2
```

```
## [1] 100   2
```

```r
dim(greenpoints) # 100 by 2
```

```
## [1] 100   2
```

```r
X=rbind(redpoints,greenpoints) # 1~100:red, 101~200:green
dim(X) # 200 by 2
```

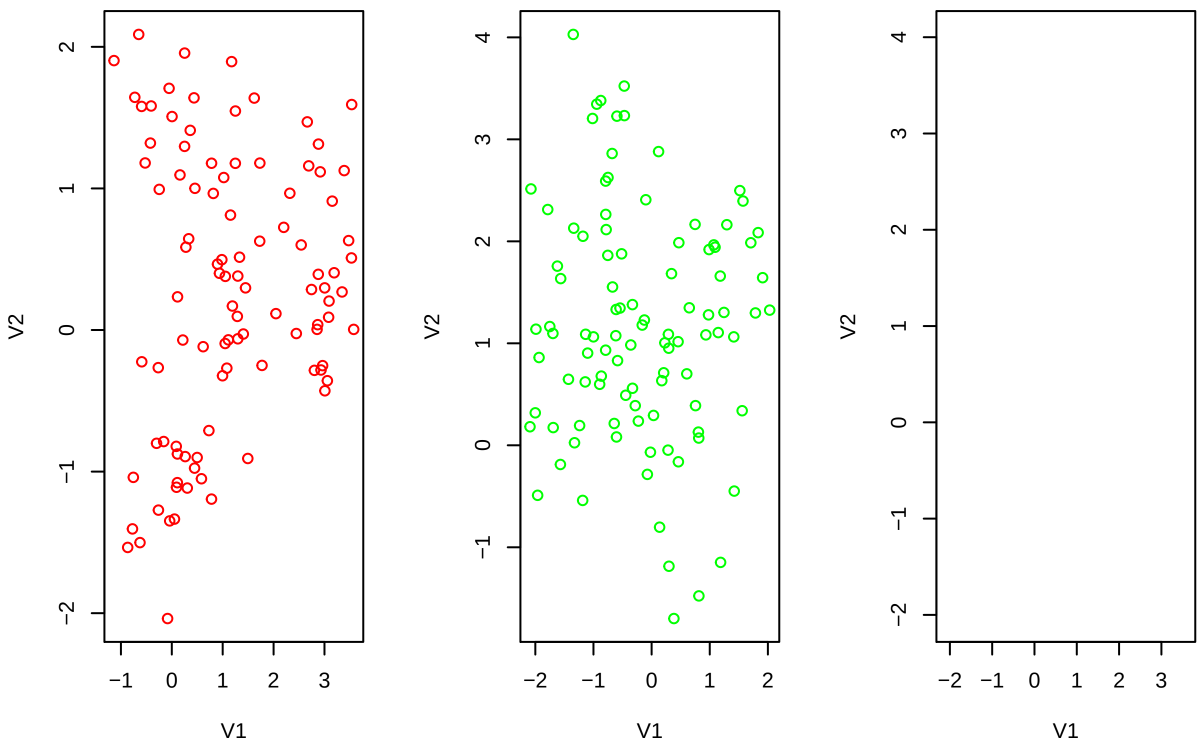
```
## [1] 200   2
```

```r
y=c(rep(1,100),rep(0,100))
y # 1~100:red(1), 101~200:green(0)
```

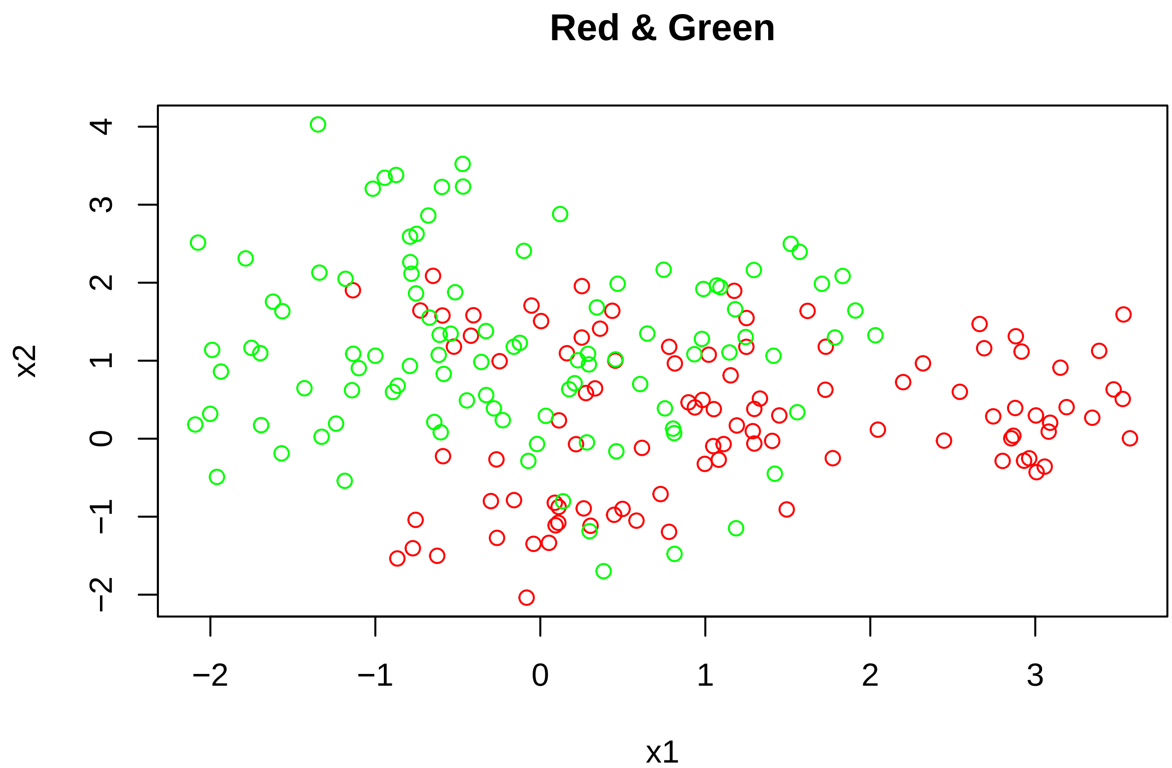
```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0
## [112] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [149] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [186] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```r
############### Q2. #################

par(mfrow=c(1,3))

plot(redpoints,col="red")
plot(greenpoints,col="green")
```

```
plot(X,type="n")
```




```
par(mfrow=c(1,1))
plot(X,type="n",main='Red & Green',xlab='x1',ylab='x2') # type="n":nothing
points(redpoints,col="red")
points(greenpoints,col="green")
```




```
############### Q3. #################

is.matrix(X)
```

```
## [1] FALSE
```

```
is.data.frame(X)
```

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

```
X=as.matrix(X)

is.matrix(X)
```

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

```
# Regression 1
X1=cbind(rep(1,nrow(X)),X) # design matrix : 200 by 3
dim(X1)
```

```
## [1] 200   3
```

```
beta.hat=solve(t(X1)%*%X1)%*%t(X1)%*%y
beta.hat
```

```
##          [,1]
##     0.5235718
## V1  0.1601129
## V2 -0.1415877
```

```r
y.hat=X1%*%beta.hat # 200 by 1

result <- ifelse(y.hat>0.5,1,0)
result[,1]
```

```
##   [1] 1 1 1 0 0 1 1 0 1 1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1 1
##  [75] 1 0 1 1 1 0 1 1 1 0 1 1 1 1 0 0 1 1 1 1 1 0 1 1 1 1 0 0 0 1 0 0 0 0 1 0 1
## [112] 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0 0
## [149] 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 1 0 1 0 0 0 0 1 1 0 1 0 0 0 0 0 0 1 0 1 0 0
## [186] 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0
```

```r
sum(result[,1]==y)
```

```
## [1] 155
```

```r
te=1-sum(result[,1]==y)/length(y); te
```

```
## [1] 0.225
```

```r
# Regression 2
fit=lm(y~X[,1]+X[,2])
summary(fit)
```

```
##
## Call:
## lm(formula = y ~ X[, 1] + X[, 2])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.87609 -0.33002  0.00249  0.30485  0.92773
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.52357    0.03653  14.332  < 2e-16 ***
## X[, 1]       0.16011    0.02124   7.539 1.69e-12 ***
## X[, 2]      -0.14159    0.02545  -5.563 8.59e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.408 on 197 degrees of freedom
## Multiple R-squared:  0.344,  Adjusted R-squared:  0.3374
## F-statistic: 51.66 on 2 and 197 DF,  p-value: < 2.2e-16
```

```r
summary(fit)$coef
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)  0.5235718 0.03653068 14.332387 2.199775e-32
## X[, 1]       0.1601129 0.02123825  7.538895 1.692278e-12
## X[, 2]      -0.1415877 0.02545322 -5.562664 8.587034e-08
```

```r
summary(fit)$coef[,1]
```

```
## (Intercept)      X[, 1]      X[, 2]
##   0.5235718   0.1601129  -0.1415877
```

```r
beta.hat2 = fit$coefficients
y.hat2 = fit$fitted.values # y.hat2 = fitted.values(fit)
```

```r
g.hat=as.numeric(y.hat>0.5) # T/F로 반환해서 numeric으로 1/0
g.hat
```

```
##   [1] 1 1 1 0 0 1 1 0 1 1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 0 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1
##  [75] 1 0 1 1 1 0 1 1 1 0 1 1 1 1 0 0 1 1 1 1 1 0 1 1 1 1 0 0 0 1 0 0 0 0 1 0 1
## [112] 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0 0
## [149] 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 1 0 1 0 0 0 0 1 1 0 1 0 0 0 0 0 0 1 0 1 0 0
## [186] 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0
```

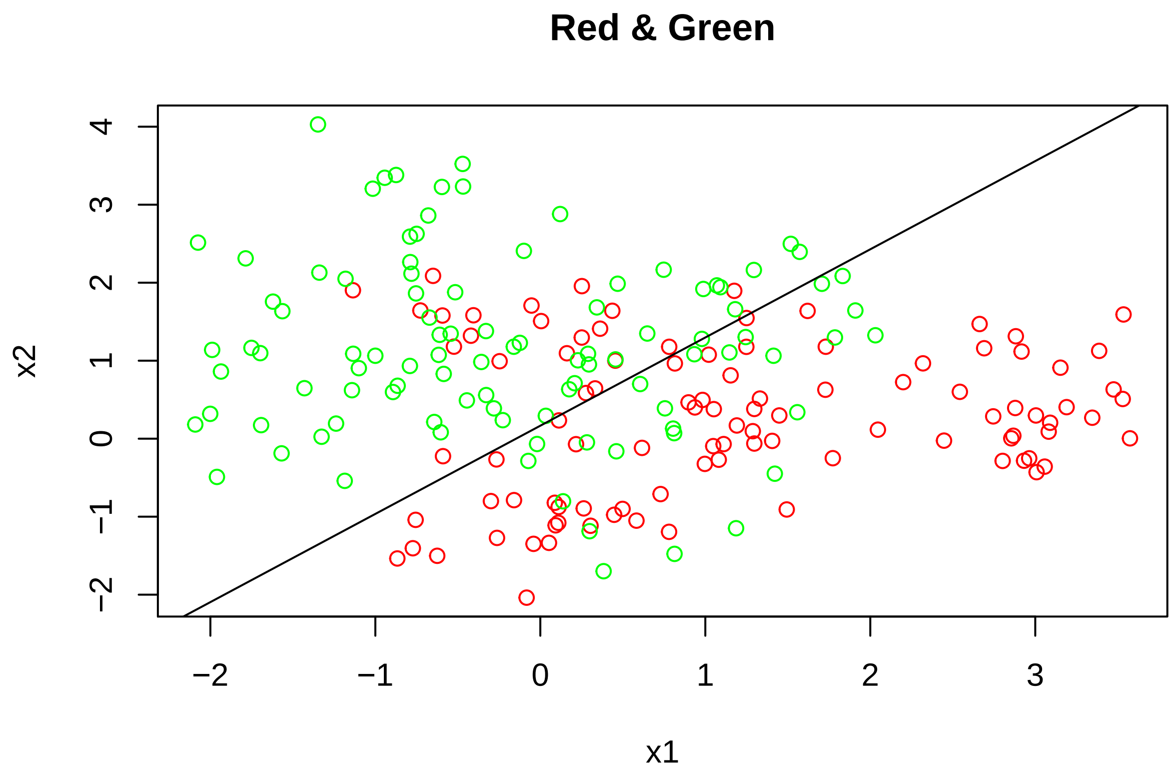
```r
z=sum(g.hat==y)
training.error=1-z/200 # 오분류율을 찾는 거니까!
training.error # 0.225
```

```
## [1] 0.225
```

```r
############### Q4. ################

plot(X,type="n",main='Red & Green',xlab='x1',ylab='x2') # type="n":nothing
points(redpoints,col="red")
points(greenpoints,col="green")
beta.hat
```

```
##           [,1]
##      0.5235718
## V1   0.1601129
## V2  -0.1415877
```

```r
abline((.5-beta.hat[1])/beta.hat[3],-beta.hat[2]/beta.hat[3])
```




```
#abline((.5-beta.hat[1,1])/beta.hat[3,1],-beta.hat[2,1]/beta.hat[3,1])

############### Q5. #################

X0=rbind(redtestpoints,greentestpoints) # testset은 각 1000 by 2
X0=as.matrix(X0) # X0는 2000 by 2

y0=c(rep(1,1000),rep(0,1000)) # 마찬가지로 y를 1~1000(red)=1, 1001~2000(green)=0
X01=cbind(rep(1,2000),X0) # design matrix(2000 by 3)

# 추정한 모수들을 이용하여 y=xb, 즉 testset에서의 예측된 색(y)을 구함, testerror
y0.hat=X01%*%beta.hat
g0.hat=as.numeric(y0.hat>0.5)

test.error=1-sum(g0.hat==y0)/2000
test.error # 0.245 > 0.225(training error)
```

```
## [1] 0.245
```

```
# testerror is always larger than training error !!!


############### Q6. #################

# Training observations 사이의 거리를 계산해야 함.
# Euclidean distance matrix
D=matrix(0,200,200) # 200 by 200 matrix all values are zero.
```

```
dim(D) # -> Nrow of training set is 200
```

```
## [1] 200 200
```

```
# Training set X is 200 by 2
for (i in 1:200) for (j in 1:200) D[i,j]=sqrt(sum((X[i,]-X[j,])^2))
# => D 200 by 200 계산으로 쭉 채움

# To find k-closest observation, Example
s=c(1,4,5,3)
sort(s) # 1 3 4 5 (ascending dataset)
```

```
## [1] 1 3 4 5
```

```
order(s) # ordered position of dataset, it return the index value
```

```
## [1] 1 4 2 3
```

```
# 1 4 2 3
# 첫 번째로 작은 값은 첫 번째 인덱스(1)에 있다.
# 두 번째로 작은 값은 네 번째 인덱스(3)에 있다.
# 세 번째로 작은 값은 두 번째 인덱스(4)에 있다.
# 네 번째로 작은 값은 세 번째 인덱스(5)에 있다.

# ex) (x1,x1)~(x1,x200)중 closest training obs 찾기
# D[1,] : (x1,x1) ~ (x1,x200)의 거리를 나타냄
order(D[1,])
```

```
##   [1]   1  23  32  74  81 114  12  49  66 130 100  35 159  31  26 152 171  34
##  [19]  95  58   2 104  67  17  77  36 192 111  44 183  51  99 160 145  40  56
##  [37] 164  63  10   3  28   9  92  70  97  72  90 181 194  22  14 153  79  61
##  [55]  13  15   7 170  84 187 109  29 185  43 166 125 197 173 110 149  38  75
##  [73] 131  47  60  46 200 188  27 140 195  37  54 169 175  89 158 126 144   4
##  [91] 128 134 174 121  87  18 117 132  71 101 147  83 184 127 163 123  59  73
## [109]  96  21  50  85  42  39  33  98  19  30 167 118 129 142  52 108 137 146
## [127] 190   6 135  45  11  48 189 112  91  93 107 199  69  80  16   5  64 102
## [145] 150 119  41  68 151 122  53  20 161 113  82 103  78 179 191 139  24  88
## [163] 156 105 115  57 138  25 136  94  65  86   8 172 141  76 116 198 154  55
## [181] 176 120 143 165 177 180 178 155 186  62 157 168 182 162 196 193 106 133
## [199] 148 124
```

```
order(D[1,])[1:7] # 7-closest training obs
```

```
## [1]   1  23  32  74  81 114  12
```

```
y # y=c(rep(1,100),rep(0,100))
```

```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0
## [112] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [149] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [186] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
y[order(D[1,])[1:7]]
```

```
## [1] 1 1 1 1 1 0 1
```

```r
mean(y[order(D[1,])[1:7]]) # y_hat value
```

```
## [1] 0.8571429
```

```r
g.hat=(mean(y[order(D[1,])[1:7]])>0.5)
g.hat
```

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

```r
# Q6.

k=7
g.hat7=rep(0,200)
for (i in 1:200) g.hat7[i]=(mean(y[order(D[i,])[1:k]])>0.5)
 g.hat7
```

```
##   [1] 1 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 1
##  [38] 1 1 0 1 1 1 0 1 0 0 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 0 1 1 1 1
##  [75] 1 0 1 1 1 0 1 1 1 0 0 1 0 1 0 0 1 1 1 1 1 0 1 1 1 1 1 0 0 0 0 0 0 0 1 0 1
## [112] 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [149] 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 1 0 0 0 0
## [186] 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0
```

```r
 training.error=1-sum(g.hat7==y)/200
 training.error
```

```
## [1] 0.2
```

```r
k=3
g.hat3=rep(0,200)
 for (i in 1:200) g.hat3[i]=(mean(y[order(D[i,])[1:k]])>0.5)
 training.error=1-sum(g.hat3==y)/200
 training.error
```

```
## [1] 0.17
```

```r
k=1
g.hat1=rep(0,200)
 for (i in 1:200) g.hat1[i]=(mean(y[order(D[i,])[1:k]])>0.5)
 training.error=1-sum(g.hat1==y)/200
 training.error
```

```
## [1] 0
```

```r
g <- NULL
for(k in c(1,3,7)){
        for(i in 1:200){
                g[i] <- mean(y[order(D[i,])[1:k]])>0.5
        }
        print(1-sum(g==y)/200)
}
```

```
## [1] 0
## [1] 0.17
## [1] 0.2
```

```r
############### Q7. #################
# Test observations 사이의 거리를 계산해야 함.
# Euclidean distance matrix
```

```r
D0=matrix(0,2000,200) # row = testset, col = trainingset
dim(D0)
```

```
## [1] 2000  200
```

```r
# (x1,x1)~(x1,x200)~...~(x2000,x1)~(x2000,x200)
for (i in 1:2000) for (j in 1:200) D0[i,j]=sqrt(sum((X0[i,]-X[j,])^2))


k=7
g0.hat=rep(0,2000)
for (i in 1:2000) g0.hat[i]=(mean(y[order(D0[i,])[1:k]])>0.5)
head(g0.hat,100)
```

```
##   [1] 1 0 1 1 0 1 1 1 0 1 1 1 0 0 1 1 1 1 0 1 1 1 0 1 0 0 0 1 1 1 0 1 0 1 0 1 1
##  [38] 1 0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1 0 0 0 1 0 1 1 1 0 1 1 1 1 0 0 1 1
##  [75] 1 1 1 0 1 1 0 1 1 1 0 1 1 0 0 0 1 0 1 0 0 1 1 1 1 1
```

```r
test.error=1-sum(g0.hat==y0)/2000
test.error
```

```
## [1] 0.2965
```

```r
k=3
g0.hat=rep(0,2000)
for (i in 1:2000) g0.hat[i]=(mean(y[order(D0[i,])[1:k]])>0.5)
head(g0.hat,100)
```

```
##   [1] 1 0 1 1 0 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 0 1 1 0 1 0 1 1
##  [38] 1 1 0 0 1 0 1 0 1 1 1 1 1 1 1 1 1 0 0 1 0 0 0 1 0 1 0 1 0 1 1 1 1 0 1 1 0
##  [75] 1 1 1 1 0 1 0 0 1 1 0 1 1 0 0 0 1 0 0 0 0 1 1 1 1 0
```

```r
test.error=1-sum(g0.hat==y0)/2000
test.error
```

```
## [1] 0.328
```

```r
k=1
g0.hat=rep(0,2000)
for (i in 1:2000) g0.hat[i]=(mean(y[order(D0[i,])[1:k]])>0.5)
head(g0.hat, 100)
```

```
##   [1] 1 1 1 1 0 1 0 1 0 1 1 1 0 0 1 0 1 1 0 1 1 1 0 1 0 0 0 0 1 1 1 1 0 0 1 0 1
##  [38] 1 0 0 0 1 1 0 0 1 1 1 1 1 1 1 0 1 0 1 1 0 0 0 1 0 1 0 0 0 1 1 1 1 0 1 1 1
##  [75] 1 1 1 1 1 1 0 1 1 1 0 1 1 0 0 0 1 0 1 0 0 1 1 1 0 1
```

```r
test.error=1-sum(g0.hat==y0)/2000
test.error
```

```
## [1] 0.3225
```

```r
g2 <- NULL
for(k in c(1,3,7)){
        for(i in 1:2000){
                g2[i] <- mean(y[order(D0[i,])[1:k]])>0.5
        }
        print(1-sum(g2==y0)/2000)
}
```

```
## [1] 0.3225
```

```
## [1] 0.328
## [1] 0.2965
```