STAT4001 Homework 1

Chan Sze Yuen Syngiene 1155127616

10/10/2021

Q1

Lets start derive it with R^2 first:

$$\begin{aligned}
R^2 &= \frac{TSS - RSS}{TSS} \\
&= \frac{\sum_{i=1}^{n}[(y_i - \bar{y})^2 - (y_i - \hat{y})^2]}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \hat{y}_i = \beta_0 + \beta_1 x_i + \varepsilon_i \text{ and } \sum_{i=1}^{n} \varepsilon_i = 0 \\
&= \frac{SS_{yy} - RSS}{SS_{yy}}
\end{aligned}$$

Recall that the OLS estimate for the β_0 and β_1:

$$\begin{aligned}
\hat{\beta}_1 &= \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2} = \frac{SS_{xy}}{SS_{xx}} \\
\hat{\beta}_0 &= \bar{y} - \hat{\beta}_1 \bar{x} = \bar{y} - \frac{SS_{xy}}{SS_{xx}}\bar{x}
\end{aligned}$$

Here we first expense the RSS (Residuals sum of squared) term:

$$\begin{aligned}
SSE &= \sum_{i=1}^{n}(y_i - \hat{y})^2 \\
&= \sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2 \\
&= \sum_{i=1}^{n}(y_i - \bar{y} + \hat{\beta}_1 \bar{x} - \beta_1 x_i)^2 \\
&= \sum_{i=1}^{n}(y_i - \bar{y} + \hat{\beta}_1(\bar{x} - x_i))^2 \\
&= SS_{yy} - 2(\frac{SS_{xy}}{SS_{xx}})SS_{xy} + (\frac{SS_{xy}}{SS_{xx}})^2 SS_{xx} \\
&= SS_{yy} - \frac{SS_{xy}^2}{SS_{xx}} \\
&= SS_{yy}(1 - \frac{SS_{xy}^2}{SS_{xx}SS_{yy}}) \\
&= SS_{yy}(1 - r_{xy}^2)
\end{aligned}$$

Hence we back to R^2:

$$\begin{aligned} R^2 &= \frac{SS_{yy} - RSS}{SS_{yy}} \\ &= \frac{SS_{yy} - SS_{yy}(1 - r_{xy}^2)}{SS_{yy}} \\ &= r_{xy}^2 \end{aligned}$$

Proved.

Q2

For the LOOCV setting, we have the number of fold=K=n=3 in this case, hence we use two sets of (x,y) as training dataset and one set of (x,y) as test dataset. Please refer to the following R code:

```
y=c(1.2,1.8,3.2)
x=c(0.4,0.8,1.2)
m=1 #LOOCV setting, only leave one set of (x,y) as test data
n=length(x)
MSE_K=c()
y_hat=c()
for (i in 1:length(y)){
slm.fit=lm(y[-i]~x[-i]) #only leave (x_i,y_i) as test data
beta=c(as.numeric(coef(slm.fit)[1]),as.numeric(coef(slm.fit)[2]))
y_hat[i]=beta[1]+beta[2]*x[i] #store y_hat
MSE_K[i]=(1/n)*(y[i]-y_hat[i])^2 #storing MSE at Kth fold
}
print(MSE_K)
```

```
## [1] 0.21333333 0.05333333 0.21333333
```

```
print(sum(MSE_K)) # LOOCV (Leave-One-Out Cross-Validation) error
```

```
## [1] 0.48
```

So the LOOCV (Leave-One-Out Cross-Validation) error is 0.48 and we are done.

Q3

Recall the defination of $\hat{\alpha}$, which minimized Var(X+Y):

$$\hat{\alpha} = \frac{\hat{\sigma_Y^2} - \sigma_{\hat{X}Y}}{\hat{\sigma_X^2} + \hat{\sigma_Y^2} - 2\sigma_{\hat{X}Y}}$$

We just use sample variance to estimate the variance here. For the bootstrap implementation, please refer to the following R code:

```
set.seed(1155127616) #for the sample function
x=c(4.3,2.1,5.3)
y=c(2.4,1.1,2.8)
alpha_hat=c()
```

```r
Bootstrap=list()
for (i in 1:3){
sample_index=sample(c(1:3),3,replace=T) #sample the index we need
if(sample_index[1]==sample_index[2] && sample_index[2]==sample_index[3]){
  sample_index=sample(c(1:3),3,replace=T)
} #prevent all the 3 samples are the same, which will cause alpha hat to be 0/0
sample_x=x[sample_index]
sample_y=y[sample_index]
Bootstrap[[i]]=matrix(c(sample_x,sample_y),byrow=F,ncol=2)
s2x=var(Bootstrap[[i]][,1])
s2y=var(Bootstrap[[i]][,2])
s2xy=cov(Bootstrap[[i]][,1],Bootstrap[[i]][,2])
alpha_hat[i]=(s2y-s2xy)/(s2x+s2y-2*s2xy)
}
print(Bootstrap) #3 sets of Bootstrap sample
```

```
## [[1]]
##      [,1] [,2]
## [1,]  2.1  1.1
## [2,]  4.3  2.4
## [3,]  5.3  2.8
##
## [[2]]
##      [,1] [,2]
## [1,]  5.3  2.8
## [2,]  2.1  1.1
## [3,]  4.3  2.4
##
## [[3]]
##      [,1] [,2]
## [1,]  2.1  1.1
## [2,]  4.3  2.4
## [3,]  4.3  2.4
```

```r
print(alpha_hat) #3 estimated alpha
```

```
## [1] -1.157895 -1.157895 -1.444444
```

```r
SE_alpha=sqrt((1/(length(alpha_hat)-1)*sum((alpha_hat-mean(alpha_hat))^2)))
print(SE_alpha)
```

```
## [1] 0.1654396
```

Hence 0.1654396 is the se we are looking for. Notice that the - sign of the alpha indicate that we should short the asset.

Q4(a)

As $\beta_0 = c_0$ is known,

$$
\begin{aligned}
L(\beta_1) &= \prod_{i=1}^{n} P(y_i \mid x_i) \\
&= \prod_{i=1}^{n} \frac{e^{(\beta_0+\beta_1 x_i)y_i}}{1+e^{\beta_0+\beta_1 x_i}} \\
l(\beta_1) &= log(L(\beta_1)) \\
&= \sum_{i=1}^{n} (\beta_0+\beta_1 x_i)y_i - \sum_{i=1}^{n} log(1+e^{\beta_0+\beta_1 x_i}) \\
l'(\beta_1) &= \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} \frac{x_i e^{\beta_0+\beta_1 x_i}}{1+e^{\beta_0+\beta_1 x_i}} \\
&= \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} \frac{x_i}{1+e^{-(\beta_0+\beta_1 x_i)}} \\
l''(\beta_1) &= \sum_{i=1}^{n} \frac{-x_i^2 e^{-(\beta_0+\beta_1 x_i)}}{(1+e^{-(\beta_0+\beta_1 x_i)})^2} \\
&= \sum_{i=1}^{n} \frac{-x_i^2}{1+e^{-(\beta_0+\beta_1 x_i)}} + \frac{x_i^2}{(1+e^{-(\beta_0+\beta_1 x_i)})^2}
\end{aligned}
$$

Q4(b)

Recall the formula of the Newton's method:

$$\beta_{1(t+1)} = \beta_{1(t)} - \frac{l'(\beta_{1(t+1)})}{l''(\beta_{1(t+1)})}$$

Which mean we want to find β_1 such that $l'(\beta_1) = 0$

Please refer to the following Rcode:

```r
x=c() ; y=c() #from data we should have these two vector
beta0=c()
beta1=1 #starting value
tol=c() #stop the loop if the difference of the updated value is smaller than tol
k=0 #count the number of loop
temp=0
L=function(x,y,beta0,beta1){prod(exp((beta0+beta1*x)*y)/(1+exp(beta0+beta1*x)))}
g=function(x,y,beta0,beta1){sum((beta0+beta1*x)*y)-sum(log(1+exp(beta0+beta1*x)))}
g1=function(x,y,beta0,beta1){sum(x*y)-sum(x/(1+exp(-(beta0+beta1*x))))}
g2=function(x,y,beta0,beta1){-sum((x^2/(1+exp(beta0+beta1*x))))+sum((x^2/(1+exp(beta0+beta1*x))^2))}
loglikelihood_trace=c(g(x,y,beta0,beta1))
beta1_trace=c(beta1)
repeat{
  temp=beta1
  beta1=beta1-g1(x,y,beta0,beta1)/g2(x,y,beta0,beta1)
  loglikelihood_trace=append(loglikelihood_trace,g(x,y,beta0,beta1))
  beta1_trace=append(beta1_trace,beta1)
  k=k+1
if(abs(L(x,y,beta0,beta1)-L(x,y,beta0,temp))<tol){
```

```
  break
}
}
```

Q4(c)

```r
setwd("D://")
load("D:/CUHKZOOMNOTESANDSOURCE/STAT4001/data/HW1Q4data1.Rdata")
head(data1) #view the data
```

```
##            x y
## 1 -0.1122575 0
## 2 -0.3115910 0
## 3  2.5828925 1
## 4  1.1134163 1
## 5  0.8358831 0
## 6  2.0022107 1
```

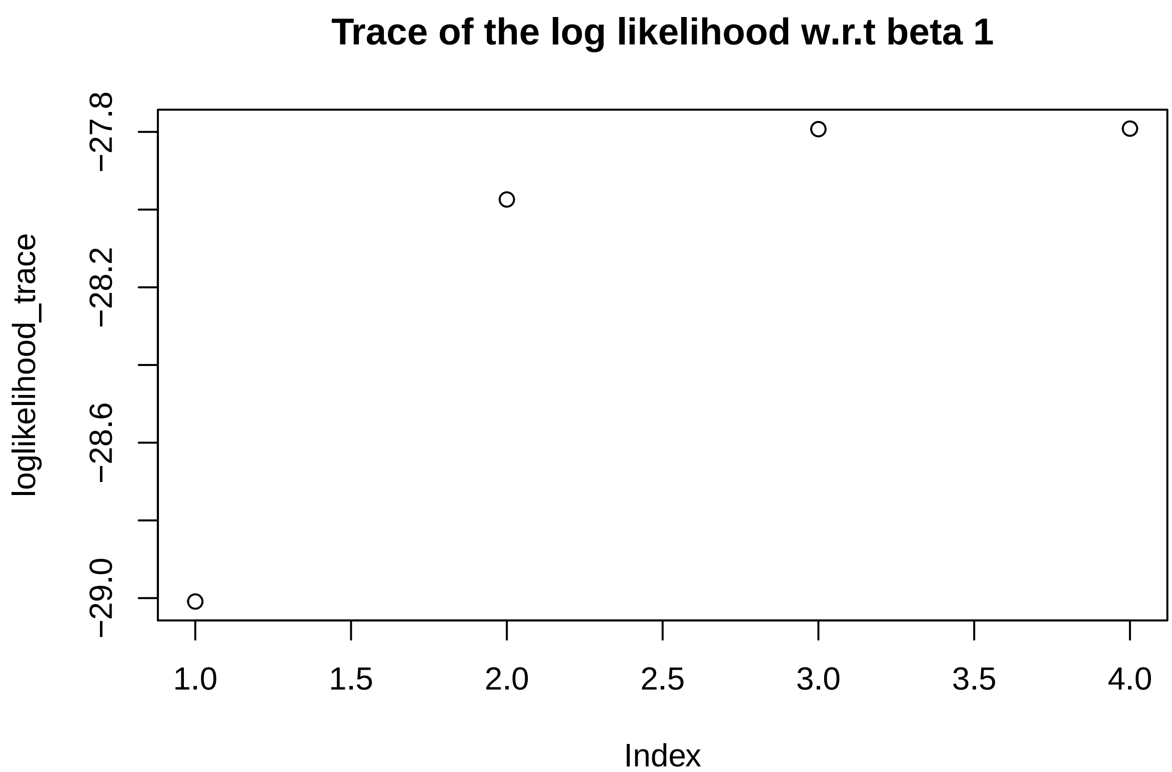
```r
x=data1$x ; y=data1$y
beta0=-0.66
beta1=1 #starting value
tol=10^-14
k=0
temp=0
L=function(x,y,beta0,beta1){prod(exp((beta0+beta1*x)*y)/(1+exp(beta0+beta1*x)))}
g=function(x,y,beta0,beta1){sum((beta0+beta1*x)*y)-sum(log(1+exp(beta0+beta1*x)))}
g1=function(x,y,beta0,beta1){sum(x*y)-sum(x/(1+exp(-(beta0+beta1*x))))}
g2=function(x,y,beta0,beta1){-sum((x^2/(1+exp(beta0+beta1*x))))+sum((x^2/(1+exp(beta0+beta1*x))^2))}
loglikelihood_trace=c(g(x,y,beta0,beta1))
beta1_trace=c(beta1)
repeat{
  temp=beta1
  beta1=beta1-g1(x,y,beta0,beta1)/g2(x,y,beta0,beta1)
  loglikelihood_trace=append(loglikelihood_trace,g(x,y,beta0,beta1))
  beta1_trace=append(beta1_trace,beta1)
  k=k+1
if(abs(L(x,y,beta0,beta1)-L(x,y,beta0,temp))<tol){
  break
}
}
loglikelihood_trace
```

```
## [1] -29.00873 -27.97400 -27.79295 -27.79161
```

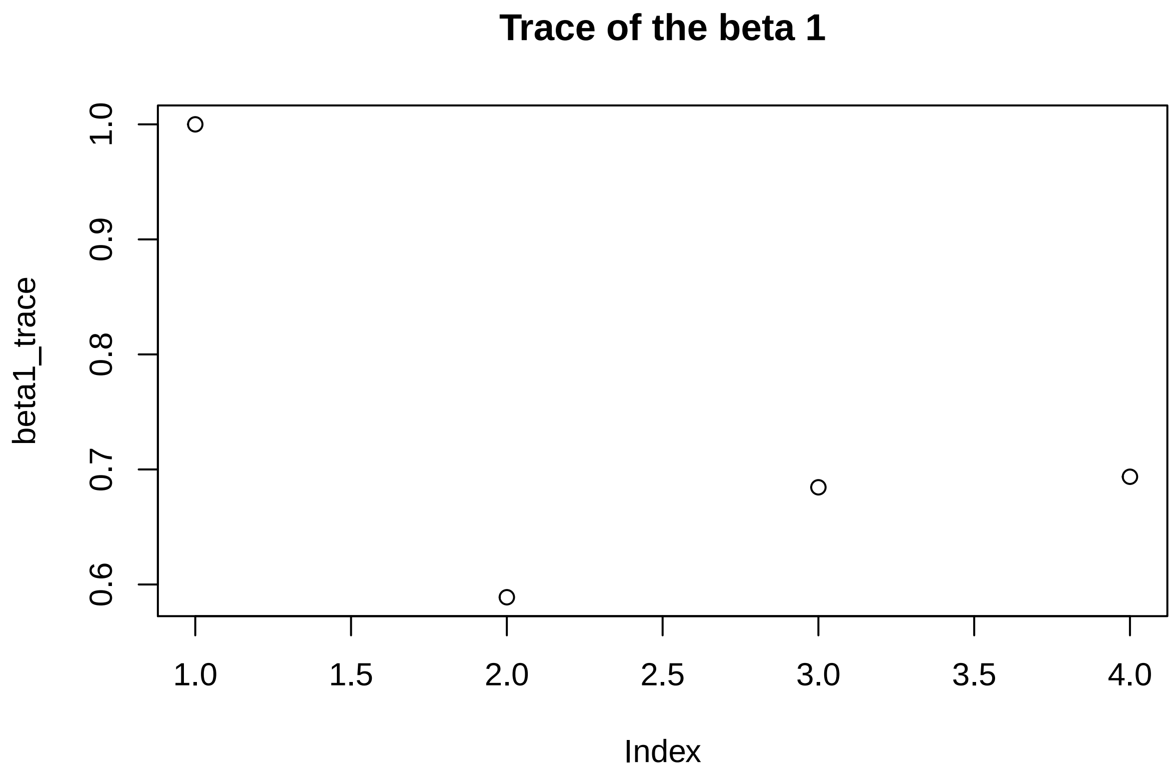
```r
beta1_trace
```

```
## [1] 1.0000000 0.5888921 0.6844710 0.6936473
```

```r
plot(loglikelihood_trace,main="Trace of the log likelihood w.r.t beta 1")
```




```r
plot(beta1_trace,main="Trace of the beta 1")
```




```r
beta1 #estimation of beta1
```

```
## [1] 0.6936473
```

```r
k #number of loop
```

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

so we have the estimate of the β_1.As we can see the likelihood of β_1 is increasing when the number of loop is increasing, also the value of the β_1 starting to be stable when the run loop is increasing.

Q4(d)

```r
setwd("D://")
load("D:/CUHKZOOMNOTESANDSOURCE/STAT4001/data/HW1Q4data2.Rdata")
head(data2) #view the data
```

```
##           x2 y2
## 1  0.3667487  1
## 2  0.6224852  1
## 3  0.2577872  1
## 4 -1.4201278  0
## 5 -2.0937814  0
## 6  1.6204320  1
```

```r
x=data2$x ; y=data2$y
beta0=0
beta1=1 #starting value
tol=10^-4
max.tol=10^10 #for stop the loop if diverges case of updating beta1 happened
k=0
temp=0
L=function(x,y,beta0,beta1){prod(exp((beta0+beta1*x)*y)/(1+exp(beta0+beta1*x)))}
g=function(x,y,beta0,beta1){sum((beta0+beta1*x)*y)-sum(log(1+exp(beta0+beta1*x)))}
g1=function(x,y,beta0,beta1){sum(x*y)-sum(x/(1+exp(-(beta0+beta1*x))))}
g2=function(x,y,beta0,beta1){-sum((x^2/(1+exp(beta0+beta1*x))))+sum((x^2/(1+exp(beta0+beta1*x))^2))}
likelihood_trace=c(g(x,y,beta0,beta1))
beta1_trace=c(beta1)
repeat{
  temp=beta1
  beta1=beta1-g1(x,y,beta0,beta1)/g2(x,y,beta0,beta1)
  likelihood_trace=append(likelihood_trace,g(x,y,beta0,beta1))
  beta1_trace=append(beta1_trace,beta1)
  k=k+1
if((abs(L(x,y,beta0,beta1)-L(x,y,beta0,temp))<tol) || ((abs(L(x,y,beta0,beta1)-L(x,y,beta0,temp)))>max.
  break
}
}
likelihood_trace
```

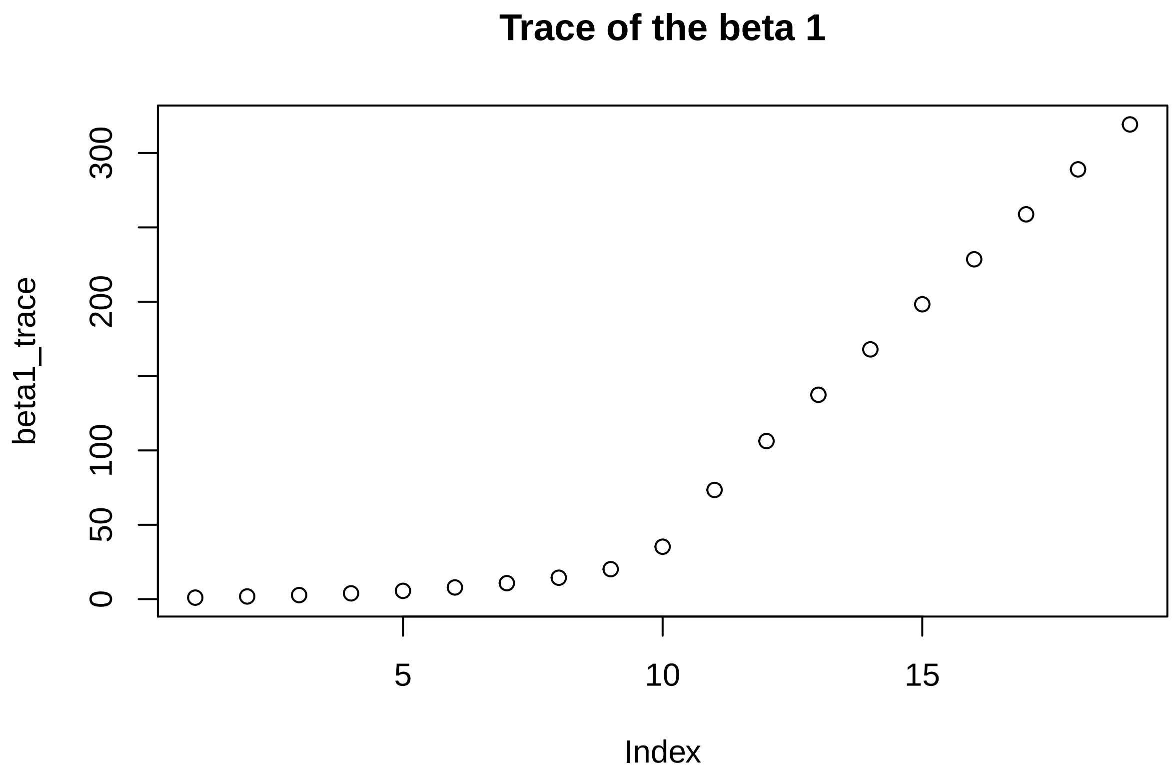
```
##  [1] -1.219173e+01 -6.405752e+00 -3.765010e+00 -2.321034e+00 -1.441229e+00
##  [6] -9.281074e-01 -6.645057e-01 -5.243561e-01 -4.219302e-01 -2.713666e-01
## [11] -8.442702e-02 -2.924334e-02 -1.054191e-02 -3.850145e-03 -1.412657e-03
## [16] -5.191853e-04 -1.909298e-04 -7.022996e-05 -2.583492e-05
```

```r
beta1_trace
```

```
##  [1]   1.000000   1.789516   2.698996   3.885114   5.558940   7.855942
##  [7]  10.730458  14.389855  20.132527  35.236068  73.411829 106.292368
## [13] 137.407730 167.946597 198.281787 228.543126 258.777439 289.001832
## [19] 319.222573
```

```r
plot(likelihood_trace,main="Trace of the log likelihood w.r.t beta 1")
```


```r
plot(beta1_trace,main="Trace of the beta 1")
```




```
beta1 #estimation of beta1
```

```
## [1] 319.2226
```

```
k #number of loop
```

```
## [1] 18
```

So clearly the β_1 diverges in this dataset, the trace of the β_1 started to increase to inf. It is notable that our likelihood also increasing in this case.

Addition: For the checking of the Deriv terms:

```
#Package for checking the Deriv term:
g1=function(x,y,beta0,beta1){sum(x*y)-sum(x/(1+exp(-(beta0+beta1*x))))}
g2=function(x,y,beta0,beta1){-sum((x^2/(1+exp(beta0+beta1*x))))+sum((x^2/(1+exp(beta0+beta1*x))^2))}
require(Deriv)
```

```
## Loading required package: Deriv
```

```
## Warning: package 'Deriv' was built under R version 3.6.3
```

```r
Deriv(g,"beta1") #first d
```

```
## function (x, y, beta0, beta1)
## {
##     .e2 <- exp(beta0 + beta1 * x)
##     sum(x * y) - sum(x * .e2/(1 + .e2))
## }
```

```r
Deriv(Deriv(g,"beta1"),"beta1") #second d
```

```
## function (x, y, beta0, beta1)
## {
##     .e2 <- exp(beta0 + beta1 * x)
##     .e3 <- 1 + .e2
##     -sum(x^2 * (1 - .e2/.e3) * .e2/.e3)
## }
```

```r
print(Deriv(g,"beta1")(x,y,1,1))==print(g1(x,y,1,1))
```

```
## [1] 16.53555
## [1] 16.53555

## [1] FALSE
```

```r
print(Deriv(Deriv(g,"beta1"),"beta1")(x,y,1,1))==print(g2(x,y,1,1))
```

```
## [1] -21.13578
## [1] -21.13578

## [1] FALSE
```

Q4(e)(i)

The model in (e) is just simply the model in (a) with $\beta_0 = c\beta_1$:

$$
\begin{aligned}
L(\beta_1) &= \prod_{i=1}^{n} P(y_i \mid x_i) \\
&= \prod_{i=1}^{n} \frac{e^{(c\beta_1 + \beta_1 x_i) y_i}}{1 + e^{c\beta_1 + \beta_1 x_i}}
\end{aligned}
$$

Using this expression we can then directly use the result in (a) by sub $\beta_0 = c\beta_1$

This is not true. Because \beta_0 is related to \beta_1, the result is not that simple.   -3

$$
\begin{aligned}
L(\beta_1) &= \prod_{i=1}^{n} P(y_i \mid x_i) \\
&= \prod_{i=1}^{n} \frac{e^{(c\beta_1+\beta_1 x_i)y_i}}{1+e^{c\beta_1+\beta_1 x_i}} \\
l(\beta_1) &= log(L(\beta_1)) \\
&= \sum_{i=1}^{n} (c\beta_1+\beta_1 x_i)y_i - \sum_{i=1}^{n} log(1+e^{c\beta_1+\beta_1 x_i}) \\
l'(\beta_1) &= \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} \frac{x_i e^{c\beta_1+\beta_1 x_i}}{1+e^{c\beta_1+\beta_1 x_i}} \\
&= \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} \frac{x_i}{1+e^{-(c\beta_1+\beta_1 x_i)}} \\
l''(\beta_1) &= \sum_{i=1}^{n} \frac{-x_i^2 e^{-(c\beta_1+\beta_1 x_i)}}{(1+e^{-(c\beta_1+\beta_1 x_i)})^2} \\
&= \sum_{i=1}^{n} \frac{-x_i^2}{1+e^{-(c\beta_1+\beta_1 x_i)}} + \frac{x_i^2}{(1+e^{-(c\beta_1+\beta_1 x_i)})^2}
\end{aligned}
$$

where$z_i = x_i + c$

Q4(e)(ii)

```r
x=c() ; y=c() ; c=c()
beta1=1 #starting value
beta0=c*beta1
tol=c()
max.tol=c() #for stop the loop if diverges of beta 1 update happen
k=0
temp=0
L=function(x,y,beta0,beta1){prod(exp((beta0+beta1*x)*y)/(1+exp(beta0+beta1*x)))}
g=function(x,y,beta0,beta1){sum((beta0+beta1*x)*y)-sum(log(1+exp(beta0+beta1*x)))}
g1=function(x,y,beta0,beta1){sum(x*y)-sum(x/(1+exp(-(beta0+beta1*x))))}
g2=function(x,y,beta0,beta1){-sum((x^2/(1+exp(beta0+beta1*x))))+sum((x^2/(1+exp(beta0+beta1*x))^2))}
likelihood_trace=c(g(x,y,beta0,beta1))
beta1_trace=c(beta1)
repeat{
  temp1=beta1
  temp0=beta0
  beta1=beta1-g1(x,y,beta0,beta1)/g2(x,y,beta0,beta1)
  beta0=c*beta1
  likelihood_trace=append(likelihood_trace,g(x,y,beta0,beta1))
  beta1_trace=append(beta1_trace,beta1)
  k=k+1
if((abs(L(x,y,beta0,beta1)-L(x,y,temp0,temp1))<tol) || ((abs(L(x,y,beta0,beta1)-L(x,y,temp0,temp1)))>ma
  break
}
}
```

Q4(e)(iii)

```
setwd("D://")
load("D:/CUHKZOOMNOTESANDSOURCE/STAT4001/data/HW1Q4data2.Rdata")
head(data2) #view the data
```

```
##           x2 y2
## 1  0.3667487  1
## 2  0.6224852  1
## 3  0.2577872  1
## 4 -1.4201278  0
## 5 -2.0937814  0
## 6  1.6204320  1
```

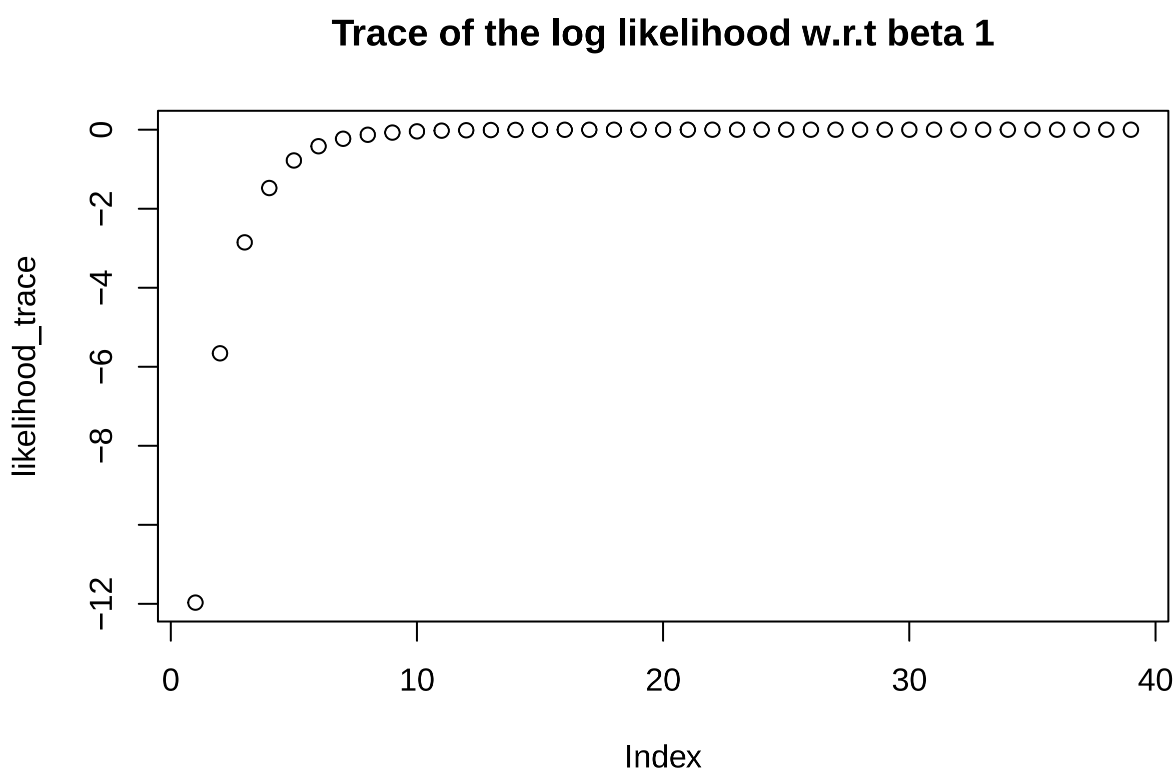
```
x=data2$x ; y=data2$y ; c=0.5
beta0=0
beta1=1 #starting value
beta0=c*beta1
tol=10^-8
max.tol=10^8 #for stop the loop if diverges of beta 1 update happen
k=0
temp=0
L=function(x,y,beta0,beta1){prod(exp((beta0+beta1*x)*y)/(1+exp(beta0+beta1*x)))}
g=function(x,y,beta0,beta1){sum((beta0+beta1*x)*y)-sum(log(1+exp(beta0+beta1*x)))}
g1=function(x,y,beta0,beta1){sum(x*y)-sum(x/(1+exp(-(beta0+beta1*x))))}
g2=function(x,y,beta0,beta1){-sum((x^2/(1+exp(beta0+beta1*x))))+sum((x^2/(1+exp(beta0+beta1*x))^2))}
likelihood_trace=c(g(x,y,beta0,beta1))
beta1_trace=c(beta1)
repeat{
  temp1=beta1
  temp0=beta0
  beta1=beta1-g1(x,y,beta0,beta1)/g2(x,y,beta0,beta1)
  beta0=c*beta1
  likelihood_trace=append(likelihood_trace,g(x,y,beta0,beta1))
  beta1_trace=append(beta1_trace,beta1)
  k=k+1
if((abs(L(x,y,beta0,beta1)-L(x,y,temp0,temp1))<tol) || ((abs(L(x,y,beta0,beta1)-L(x,y,temp0,temp1)))>ma
  break
}
}
likelihood_trace
```

```
##  [1] -1.196871e+01 -5.659085e+00 -2.851086e+00 -1.475739e+00 -7.785787e-01
##  [6] -4.183852e-01 -2.291267e-01 -1.277950e-01 -7.243362e-02 -4.159829e-02
## [11] -2.413494e-02 -1.411111e-02 -8.297710e-03 -4.899890e-03 -2.902455e-03
## [16] -1.723241e-03 -1.024885e-03 -6.103372e-04 -3.638293e-04 -2.170513e-04
## [21] -1.295668e-04 -7.738204e-05 -4.623424e-05 -2.763352e-05 -1.652100e-05
## [26] -9.879815e-06 -5.909643e-06 -3.535613e-06 -2.115690e-06 -1.266246e-06
## [31] -7.579811e-07 -4.538052e-07 -2.717371e-07 -1.627400e-07 -9.747737e-08
## [36] -5.839502e-08 -3.498735e-08 -2.096544e-08 -1.256467e-08
```

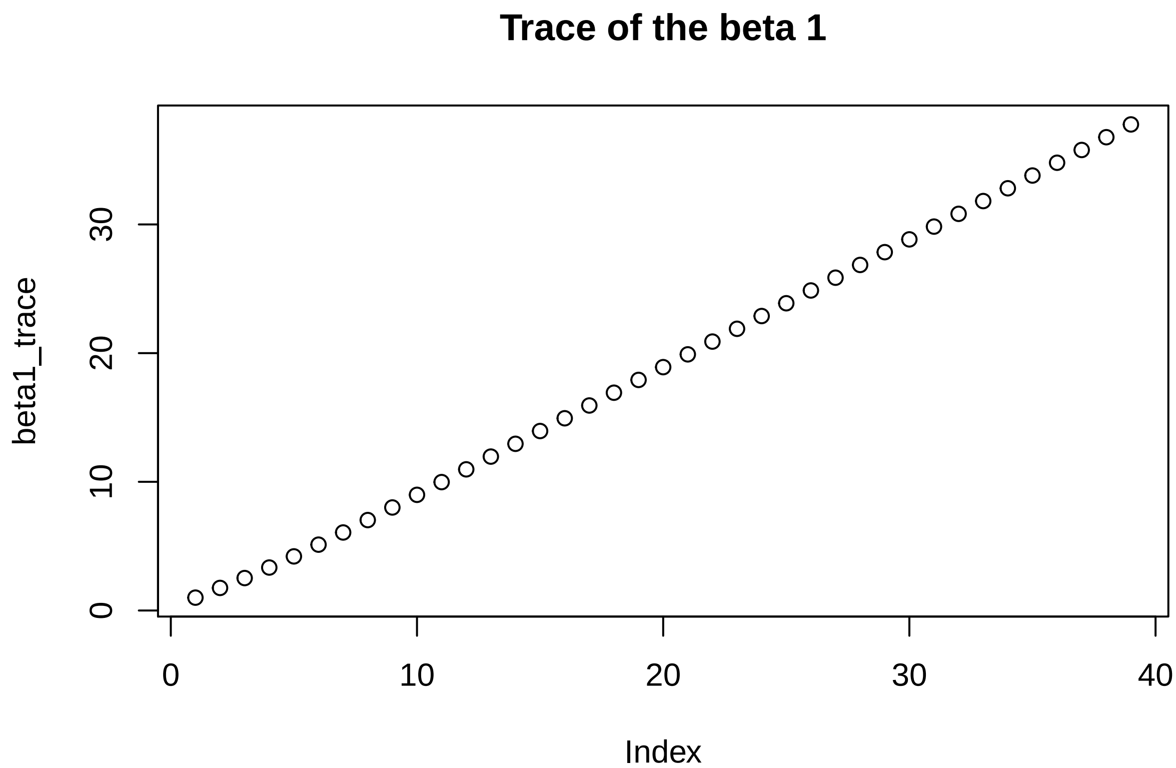
```
beta1_trace
```

```
##  [1]  1.000000  1.754206  2.522126  3.338310  4.206039  5.117874  6.062062
##  [8]  7.027210  8.004797  8.989404  9.977888 10.968504 11.960296 12.952733
## [15] 13.945517 14.938479 15.931520 16.924582 17.917632 18.910652 19.903632
## [22] 20.896568 21.889458 22.882304 23.875106 24.867867 25.860588 26.853273
## [29] 27.845922 28.838539 29.831124 30.823680 31.816207 32.808707 33.801181
## [36] 34.793630 35.786055 36.778456 37.770835
```

```
plot(likelihood_trace,main="Trace of the log likelihood w.r.t beta 1")
```




```
plot(beta1_trace,main="Trace of the beta 1")
```




```r
beta1 #estimation of beta1
```

```
## [1] 37.77084
```

```r
k #number of loop
```

```
## [1] 38
```

As shown above, the update of $beta_1$ is still diverges.

Q4(f)

Decision boundary for model in (d):

$$\frac{e^{(\beta_0+\beta_1 x_i)y_i}}{1+e^{\beta_0+\beta_1 x_i}} = 0.5$$
$$x_i = 0$$

Decision boundary for model in (e):

$$\frac{e^{(c\beta_1+\beta_1 x_i)y_i}}{1+e^{c\beta_1+\beta_1 x_i}} = 0.5$$
$$x_i = -c = -0.5$$

Q5

i will combine both a and b in this question:

```
KNN=function(Q5data,n,p,k,x,y,x_new){
dist=c()
for (i in c(1:n)){
  dist=append(dist,sqrt(sum(x[i,]-x_new)^2))
}
dist_sort=order(dist)
vote=c()
for (i in 1:k){
  vote=append(vote,y[dist_sort[i]])
}
result=c()
if (mean(vote)<0.5){
  result=rep(0,length(x_new))
  }else{
  result=rep(1,length(x_new))
  }
cat("Vote:",vote,"\nx_new:",x_new,"\nPrediction:",result,"\n")
}
setwd("D://")
Q5data=get(load("D:/CUHKZOOMNOTESANDSOURCE/STAT4001/data/HW1Q5data.Rdata"))
x=Q5data$x; y=Q5data$y ; x_new=Q5data$x_new
n=dim(x)[1]
p=dim(x)[2]
k=8
KNN(Q5data,n,p,k,x,y,x_new)
```

sum((x[i,]-x_new)^2)
You should take square before summation.

-3

```
## Vote: 1 0 1 1 1 1 0 1
## x_new: 0.9091746 0.0651728 0.1222822 0.5869917 -0.7918291 -0.2188081 -0.4995232 0.01360999 0.6955275
## Prediction: 1 1 1 1 1 1 1 1 1 1
```

Q6

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 3.6.3
```

```
names(Weekly)
```

```
## [1] "Year"      "Lag1"      "Lag2"      "Lag3"      "Lag4"      "Lag5"
## [7] "Volume"    "Today"     "Direction"
```

```
head(Weekly)
```

```
##   Year  Lag1   Lag2   Lag3   Lag4   Lag5    Volume  Today Direction
## 1 1990  0.816  1.572 -3.936 -0.229 -3.484 0.1549760 -0.270      Down
```

```
## 2 1990 -0.270  0.816  1.572 -3.936 -0.229 0.1485740 -2.576      Down
## 3 1990 -2.576 -0.270  0.816  1.572 -3.936 0.1598375  3.514        Up
## 4 1990  3.514 -2.576 -0.270  0.816  1.572 0.1616300  0.712        Up
## 5 1990  0.712  3.514 -2.576 -0.270  0.816 0.1537280  1.178        Up
## 6 1990  1.178  0.712  3.514 -2.576 -0.270 0.1544440 -1.372      Down
```

```
#(a) Logistic Regression
glm.fit=glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Weekly,family=binomial)
summary(glm.fit)$coef
```

```
##                Estimate Std. Error    z value    Pr(>|z|)
## (Intercept)  0.26686414 0.08592961  3.1056134 0.001898848
## Lag1        -0.04126894 0.02641026 -1.5626099 0.118144368
## Lag2         0.05844168 0.02686499  2.1753839 0.029601361
## Lag3        -0.01606114 0.02666299 -0.6023760 0.546923890
## Lag4        -0.02779021 0.02646332 -1.0501409 0.293653342
## Lag5        -0.01447206 0.02638478 -0.5485006 0.583348244
## Volume      -0.02274153 0.03689812 -0.6163330 0.537674762
```

```
summary(glm.fit)$coef[,4]
```

```
## (Intercept)        Lag1        Lag2        Lag3        Lag4        Lag5
## 0.001898848 0.118144368 0.029601361 0.546923890 0.293653342 0.583348244
##      Volume
## 0.537674762
```

```
glm.probs=predict(glm.fit,type="response")
glm.predict=rep("Down",dim(Weekly)[1])
glm.predict[glm.probs>0.5]="Up"
glm.predict[glm.probs<0.5]="Down"
Act.Direction=Weekly$Direction
confusion_matrix=matrix(c(length(Act.Direction[Act.Direction=="Up"]),
                          length(Act.Direction[Act.Direction=="Down"]),
                          length(glm.predict[glm.predict=="Up"]),
                          length(glm.predict[glm.predict=="Down"])),
                        byrow=T,ncol=2)
colnames(confusion_matrix)=c("Actual:Up","Actual:Down")
rownames(confusion_matrix)=c("Predict:Up","Predict:Down")
confusion_matrix
```

```
##              Actual:Up Actual:Down
## Predict:Up         605         484
## Predict:Down       987         102
```

```
CR_logit=(confusion_matrix[1,1]+confusion_matrix[2,2])/dim(Weekly)[1]
CR_logit #rate of predict correctly
```

```
## [1] 0.6492195
```

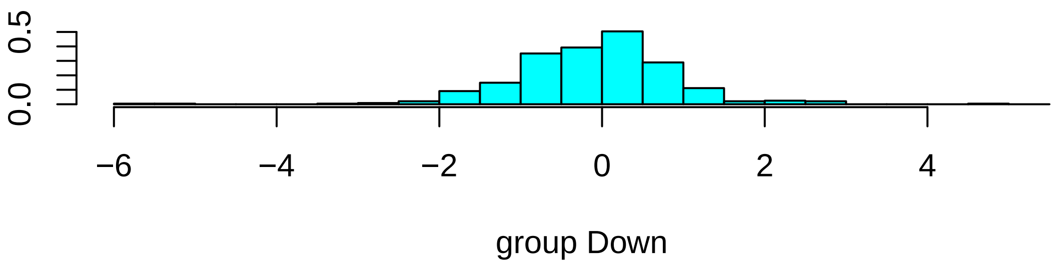
```r
#(b) Linear Discriminant Analysis
library(ISLR)
library(MASS)
```

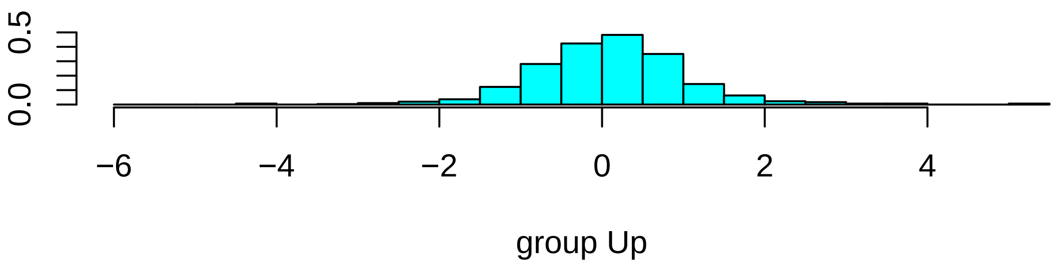
```
## Warning: package 'MASS' was built under R version 3.6.3
```

```r
lda.fit=lda(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Weekly)
lda.fit
```

```
## Call:
## lda(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4444444 0.5555556
##
## Group means:
##            Lag1        Lag2       Lag3      Lag4      Lag5   Volume
## Down 0.28229545 -0.04042355 0.20764669 0.2000207 0.1878347 1.608536
## Up   0.04521653  0.30428099 0.09885124 0.1024562 0.1015388 1.547483
##
## Coefficients of linear discriminants:
##                LD1
## Lag1   -0.21451867
## Lag2    0.30090869
## Lag3   -0.08015487
## Lag4   -0.14217986
## Lag5   -0.07271067
## Volume -0.12269898
```

```r
plot(lda.fit)
```






```
lda.pred=predict(lda.fit, Weekly)
names(lda.pred)
```

```
## [1] "class"     "posterior" "x"
```

```
lda.class=lda.pred$class
table_LDA=table(lda.class,Act.Direction) ;print(table_LDA)
```

```
##          Act.Direction
## lda.class Down  Up
##      Down   52  46
##      Up    432 559
```

```
CR_LDA=(table_LDA[1,1]+table_LDA[2,2])/dim(Weekly)[1]
CR_LDA #rate of predict correctly
```

```
## [1] 0.5610652
```

```
#(c) Quadratic Discriminant Analysis
qda.fit=qda(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,data=Weekly)
qda.fit
```

```
## Call:
```

```
## qda(Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 + Volume, data = Weekly)
##
## Prior probabilities of groups:
##      Down        Up
## 0.4444444 0.5555556
##
## Group means:
##            Lag1        Lag2       Lag3      Lag4      Lag5   Volume
## Down 0.28229545 -0.04042355 0.20764669 0.2000207 0.1878347 1.608536
## Up   0.04521653  0.30428099 0.09885124 0.1024562 0.1015388 1.547483
```

```
qda.class=predict(qda.fit,Weekly)$class
table_QDA=table(qda.class,Act.Direction) ;print(table_QDA)
```

```
##          Act.Direction
## qda.class Down  Up
##      Down  127 119
##      Up    357 486
```

```
CR_QDA=(table_QDA[1,1]+table_QDA[2,2])/dim(Weekly)[1]
CR_QDA #rate of predict correctly
```

```
## [1] 0.5629017
```

```
#(d) K-Nearest Neighbors
library(class)
```

```
## Warning: package 'class' was built under R version 3.6.3
```

```
set.seed(1)
traindata=subset(Weekly,Year<2005,select=c("Lag1","Lag2","Lag3","Lag4","Lag5","Volume"))
testdata=subset(Weekly,Year>2004,select=c("Lag1","Lag2","Lag3","Lag4","Lag5","Volume"))
test_tr=subset(Weekly,Year>2004,select=c("Direction"))
train_tr=subset(Weekly,Year<2005,select=c("Direction"))
knn.pred=knn(train=traindata,test=testdata,cl=train_tr[,1],k=1)
table_KNN=table(knn.pred,test_tr[,1])
CR_KNN=(table_KNN[1,1]+table_KNN[2,2])/length(knn.pred)
CR_KNN #rate of predict correctly
```

```
## [1] 0.4760383
```

```
CR=c(CR_logit,CR_LDA,CR_QDA,CR_KNN) ; names(CR)=c("CR_logit","CR_LDA","CR_QDA","CR_KNN"); print(CR)
```

```
##  CR_logit    CR_LDA    CR_QDA    CR_KNN
## 0.6492195 0.5610652 0.5629017 0.4760383
```

Hence logit regression give us highest correctly predict rate.