> #Exercise1 Data creation

> x1<-as.vector(runif(10000,1,3))# Generate a vector of a uniform distribution

> x2<-as.vector(rgamma(10000,3,2))# Generate a vector of a gamma distribution

> x3<-as.vector(rbinom(10000,1,0.3))#Generate a vector of a binomial distribution

> eps<-as.vector(rnorm(10000,2,1))# Generate a vector of a normal distribution

> Y<-as.vector(0.5+1.2\*x1-0.9\*x2+0.1\*x3+eps)# Generate Y variable

> ydum<-as.vector(rep(0,10000)) # Generate a vector of 10000 zeros

> for (i in 1:10000) {

+ if( Y[i]> mean(Y)){

+ ydum[i]=1

+ }

+ }# Generate ydum.

>

>

> #Exercise2 OLS

> #2.1

> cor.test(Y,x1)

Pearson's product-moment correlation

data: Y and x1

t = 55.366, df = 9998, p-value < 2.2e-16

alternative hypothesis: true correlation is not equal to 0

95 percent confidence interval:

0.4692708 0.4992750

sample estimates:

cor

0.4844153

> # The correlation between Y and x1 is 0.4967823.

> #The difference between the correlation and 1.2 is because Y are also affected by x2 and x3.

> #2.3

> x0<-as.vector(rep(1,10000))

> X<-matrix(c(x0,x1,x2,x3),10000,4)

> beta<-as.vector(solve(t(X)%\*%X)%\*%t(X)%\*%Y)

> # The coefficients on this regression

> #2.4

> #Caculate the standard errors

> #Using the formula

> beta<-matrix(c(beta),4,1)

> res<-Y-X%\*%beta

> s2<-t(res)%\*%res/9995

> #Using bootstrap with 49 and 499 replications respectively

> B<-49

> betasample<-as.data.frame(matrix(numeric(0),ncol = 4))

> for (i in 1:B) {

+ data <- cbind(Y,X)

+ sampleX<- data[sample(10000,10000,replace = T),]

+ beta<-matrix(c(solve(t(sampleX[,2:5])%\*%sampleX[,2:5])%\*%t(sampleX[,2:5])%\*%sampleX[,1]),1,4)

+ betasample<-rbind(betasample,beta)

+ }

> seboot1<-data.frame(apply(betasample, 2,sd))

>

> B<-499

> betasample<-as.data.frame(matrix(numeric(0),ncol = 4))

> for (i in 1:B) {

+ data <- cbind(Y,X)

+ sampleX<- data[sample(10000,10000,replace = T),]

+ beta<-matrix(c(solve(t(sampleX[,2:5])%\*%sampleX[,2:5])%\*%t(sampleX[,2:5])%\*%sampleX[,1]),1,4)

+ betasample<-rbind(betasample,beta)

+ }

> seboot2<-apply(betasample, 2,sd)

> #Exercise 3

> #3.1

> ydum<-matrix(ydum,10000,1)

> beta\_func<-function(beta\_probit){

+ return(sum(ydum\*log(pnorm(X%\*%beta\_probit)))+sum((1-ydum)\*log(1-pnorm(X%\*%beta\_probit))))

+ }# Generate the likelihood function

> # test:a = matrix(c(0.5, 0.5, 0.5, 0.3), 4, 1)

> # beta\_func(a)

> #3.2

> i<-matrix(c(0.5, 0.5, 0.5, 0.3), 4, 1)

> a<-diag(0.00001,4,4)

> i\_new<-matrix(c(0.49999,0.49999,0.49999,0.29999),4,1)#Generate the start point of beta

> while (beta\_func(i)<beta\_func(i\_new)) {

+ i<-i\_new

+ i\_combo<-matrix(c(rep(i,4)),4,4)

+ i\_combo\_new<-i\_combo+a

+ b0<-(beta\_func(i\_combo\_new[,1])-beta\_func(i\_combo[,1]))/0.00001

+ b1<-(beta\_func(i\_combo\_new[,2])-beta\_func(i\_combo[,2]))/0.00001

+ b2<-(beta\_func(i\_combo\_new[,3])-beta\_func(i\_combo[,3]))/0.00001

+ b3<-(beta\_func(i\_combo\_new[,4])-beta\_func(i\_combo[,4]))/0.00001

+ dk<-matrix(c(b0,b1,b2,b3),4,1)

+ i\_new<-i+0.00001\*dk

+ }

> print(i)

[,1]

[1,] -1.06932628

[2,] 1.19573664

[3,] -0.89398683

[4,] 0.06264785

> # The i when the loop stops is the maximum likelihood

> #3.3

> #The difference between the result from 3.2 and beta is in the coeffience of x0=0.5

> # There is no difference among other coefficients

>

> #Exercise 4

> probit<-glm(ydum~0+X, family = binomial(link = "probit"))

> logit<-glm(ydum~0+X, family = binomial(link = "logit"))

> linear<-lm(ydum~0+X)

> beta\_probit<-as.vector(probit$coefficients)

> beta\_logit<-as.vector(logit$coefficients)

> beta\_linear<-as.vector(linear$coefficients)

> beta\_compare<-data.frame(beta\_probit,beta\_logit,beta\_linear)

> summary(probit)

Call:

glm(formula = ydum ~ 0 + X, family = binomial(link = "probit"))

Deviance Residuals:

Min 1Q Median 3Q Max

-2.8439 -0.8163 0.2232 0.8026 2.8442

Coefficients:

Estimate Std. Error z value Pr(>|z|)

X1 -1.06940 0.05711 -18.726 <2e-16 \*\*\*

X2 1.19577 0.02776 43.075 <2e-16 \*\*\*

X3 -0.89398 0.02139 -41.794 <2e-16 \*\*\*

X4 0.06265 0.03159 1.983 0.0474 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 13862.9 on 10000 degrees of freedom

Residual deviance: 9818.7 on 9996 degrees of freedom

AIC: 9826.7

Number of Fisher Scoring iterations: 5

> summary(logit)

Call:

glm(formula = ydum ~ 0 + X, family = binomial(link = "logit"))

Deviance Residuals:

Min 1Q Median 3Q Max

-2.6869 -0.7994 0.2641 0.7874 2.6964

Coefficients:

Estimate Std. Error z value Pr(>|z|)

X1 -1.80294 0.09688 -18.611 <2e-16 \*\*\*

X2 2.02587 0.04985 40.641 <2e-16 \*\*\*

X3 -1.52417 0.03871 -39.370 <2e-16 \*\*\*

X4 0.11055 0.05389 2.051 0.0402 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 13862.9 on 10000 degrees of freedom

Residual deviance: 9825.7 on 9996 degrees of freedom

AIC: 9833.7

Number of Fisher Scoring iterations: 5

> summary(linear)

Call:

lm(formula = ydum ~ 0 + X)

Residuals:

Min 1Q Median 3Q Max

-1.09132 -0.34581 0.04305 0.32915 1.03737

Coefficients:

Estimate Std. Error t value Pr(>|t|)

X1 0.154267 0.016590 9.299 <2e-16 \*\*\*

X2 0.350474 0.007089 49.442 <2e-16 \*\*\*

X3 -0.233583 0.004754 -49.137 <2e-16 \*\*\*

X4 0.019493 0.008898 2.191 0.0285 \*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4094 on 9996 degrees of freedom

Multiple R-squared: 0.671, Adjusted R-squared: 0.6708

F-statistic: 5096 on 4 and 9996 DF, p-value: < 2.2e-16

> # There are literally no difference of coefficients between logit and probit models.

>

> #Exercise 5

> #5.1

> Xbeta<-mean(X%\*%beta\_probit)

> fideriv<-(pnorm(Xbeta+0.00001)-pnorm(Xbeta))/0.00001

> margief\_probit<-data.frame(fideriv\*beta\_probit)

>

> # The outcome of marginal effects of probit models

> Xbeta2<-mean(X%\*%beta\_logit)

> logit\_func<-exp(Xbeta2)/(1+exp(Xbeta2))

> logit\_func1<-exp(Xbeta2+0.00001)/(1+exp(Xbeta2+0.00001))

> fideriv2<-(logit\_func1-logit\_func)/0.00001

> margief\_logit<-data.frame(fideriv2\*beta\_logit)

> # The outcome of marginal effects of logit models

> #5.2

> #Using delta method to compute the standard deviations.

> V\_probit <- vcov(probit)

> beta\_probit\_new<-data.frame(beta\_probit,beta\_probit,beta\_probit,beta\_probit)

> beta\_probit\_new<-beta\_probit\_new+diag(0.00001,4)

> M1 <- function(beta\_probit) mean(dnorm(X%\*%beta\_probit)\*beta\_probit[1])

> g11<-(M1(beta\_probit\_new[,1])-M1(beta\_probit))/0.00001

> g12<-(M1(beta\_probit\_new[,2])-M1(beta\_probit))/0.00001

> g13<-(M1(beta\_probit\_new[,3])-M1(beta\_probit))/0.00001

> g14<-(M1(beta\_probit\_new[,4])-M1(beta\_probit))/0.00001

> g1<-matrix(c(g11,g12,g13,g14),1,4)

> # The first row of Jacobian matrix

> M2 <- function(beta\_probit) mean(dnorm(X%\*%beta\_probit)\*beta\_probit[2])

> g21<-(M2(beta\_probit\_new[,1])-M2(beta\_probit))/0.00001

> g22<-(M2(beta\_probit\_new[,2])-M2(beta\_probit))/0.00001

> g23<-(M2(beta\_probit\_new[,3])-M2(beta\_probit))/0.00001

> g24<-(M2(beta\_probit\_new[,4])-M2(beta\_probit))/0.00001

> g2<-matrix(c(g21,g22,g23,g24),1,4)

> M3 <- function(beta\_probit) mean(dnorm(X%\*%beta\_probit)\*beta\_probit[3])

> g31<-(M3(beta\_probit\_new[,1])-M3(beta\_probit))/0.00001

> g32<-(M3(beta\_probit\_new[,2])-M3(beta\_probit))/0.00001

> g33<-(M3(beta\_probit\_new[,3])-M3(beta\_probit))/0.00001

> g34<-(M3(beta\_probit\_new[,4])-M3(beta\_probit))/0.00001

> g3<-matrix(c(g31,g32,g33,g34),1,4)

> M4 <- function(beta\_probit) mean(dnorm(X%\*%beta\_probit)\*beta\_probit[4])

> g41<-(M4(beta\_probit\_new[,1])-M4(beta\_probit))/0.00001

> g42<-(M4(beta\_probit\_new[,2])-M4(beta\_probit))/0.00001

> g43<-(M4(beta\_probit\_new[,3])-M4(beta\_probit))/0.00001

> g44<-(M4(beta\_probit\_new[,4])-M4(beta\_probit))/0.00001

> g4<-matrix(c(g41,g42,g43,g44),1,4)

>

> J\_probit<-rbind(g1,g2,g3,g4)

> delta\_probit<-t(J\_probit)%\*%V\_probit%\*%J\_probit# The result of probit model

> sd\_probit<-diag(delta\_probit)

> V\_logit <- vcov(logit)

> beta\_logit\_new<-data.frame(beta\_logit,beta\_logit,beta\_logit,beta\_logit)

> beta\_logit\_new<-beta\_logit\_new+diag(0.00001,4)

> M1 <- function(beta\_logit) mean(dlogis(X%\*%beta\_logit)\*beta\_logit[1])

> g11<-(M1(beta\_logit\_new[,1])-M1(beta\_logit))/0.00001

> g12<-(M1(beta\_logit\_new[,2])-M1(beta\_logit))/0.00001

> g13<-(M1(beta\_logit\_new[,3])-M1(beta\_logit))/0.00001

> g14<-(M1(beta\_logit\_new[,4])-M1(beta\_logit))/0.00001

> g1<-matrix(c(g11,g12,g13,g14),1,4)

> # The first row of Jacobian matrix

> M2 <- function(beta\_logit) mean(dlogis(X%\*%beta\_logit)\*beta\_logit[2])

> g21<-(M2(beta\_logit\_new[,1])-M2(beta\_logit))/0.00001

> g22<-(M2(beta\_logit\_new[,2])-M2(beta\_logit))/0.00001

> g23<-(M2(beta\_logit\_new[,3])-M2(beta\_logit))/0.00001

> g24<-(M2(beta\_logit\_new[,4])-M2(beta\_logit))/0.00001

> g2<-matrix(c(g21,g22,g23,g24),1,4)

> M3 <- function(beta\_logit) mean(dlogis(X%\*%beta\_logit)\*beta\_logit[3])

> g31<-(M3(beta\_logit\_new[,1])-M3(beta\_logit))/0.00001

> g32<-(M3(beta\_logit\_new[,2])-M3(beta\_logit))/0.00001

> g33<-(M3(beta\_logit\_new[,3])-M3(beta\_logit))/0.00001

> g34<-(M3(beta\_logit\_new[,4])-M3(beta\_logit))/0.00001

> g3<-matrix(c(g31,g32,g33,g34),1,4)

> M4 <- function(beta\_logit) mean(dlogis(X%\*%beta\_logit)\*beta\_logit[4])

> g41<-(M4(beta\_logit\_new[,1])-M4(beta\_logit))/0.00001

> g42<-(M4(beta\_logit\_new[,2])-M4(beta\_logit))/0.00001

> g43<-(M4(beta\_logit\_new[,3])-M4(beta\_logit))/0.00001

> g44<-(M4(beta\_logit\_new[,4])-M4(beta\_logit))/0.00001

> g4<-matrix(c(g41,g42,g43,g44),1,4)

>

> J\_logit<-rbind(g1,g2,g3,g4)

> delta\_logit<-t(J\_logit)%\*%V\_logit%\*%J\_logit

> sd\_logit<-diag(delta\_logit)

>

>

> #Using bootstrap to compute the standard errors

> B<-499

> mesample<-as.data.frame(matrix(numeric(0),ncol = 4))

> for (i in 1:B) {

+ data <- cbind(ydum,X)

+ sampleX<- data[sample(10000,10000,replace = T),]

+ probitboot<-glm(sampleX[,1]~0+sampleX[,2:5], family = binomial(link = "probit"))

+ coeboot<-probitboot$coefficients

+ meanboot<-mean(sampleX[,2:5]%\*%coeboot)

+ fiderivboot<-(pnorm(meanboot+0.00001)-pnorm(meanboot))/0.00001

+ margief<-matrix(fideriv\*coeboot,1,4)

+ mesample<-rbind(mesample, margief)

+ }

> mead<-apply(mesample,2,sd)

> # The standard deviations of probit model

> B<-499

> mesample<-as.data.frame(matrix(numeric(0),ncol = 4))

> for (i in 1:B) {

+ data <- cbind(ydum,X)

+ sampleX<- data[sample(10000,10000,replace = T),]

+ probitboot<-glm(sampleX[,1]~0+sampleX[,2:5], family = binomial(link = "logit"))

+ coeboot<-probitboot$coefficients

+ meanboot<-mean(sampleX[,2:5]%\*%coeboot)

+ fiderivboot<-(pnorm(meanboot+0.00001)-pnorm(meanboot))/0.00001

+ margief<-matrix(fideriv\*coeboot,1,4)

+ mesample<-rbind(mesample, margief)

+ }

> mead2<-apply(mesample,2,sd)

> # The standard deviations of logit model