Logistic Regression

- **Regression Models with Binary Response Variable.**

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i \qquad Y_i = 0, 1$$

Note: If $E(\epsilon_i) = 0$, then $E(Y_i) = \beta_0 + \beta_1 X_i = \pi_i$.

- Special Problems:
  1. Nonnormal Error Terms:
     - When $Y_i = 1$, $\epsilon = 1 - \beta_0 - \beta_1 X_1$.
     - When $Y_i = 0$, $\epsilon = 0 - \beta_0 - \beta_1 X_1$.
  2. Nonconstant Error Variance: $V(Y_i) = \pi_i(1 - \pi_i)$
  3. Constraints on Response Function: $0 \leq E(Y) \leq 1$

- **Sigmoidal Response Function for Binary Responses.**
  1. Probit Model: $E(Y_i) = \pi_i = \Phi(\beta_0 + \beta_1 X_i)$
  2. Logistic Model: $E(Y_i) = \pi_i = \dfrac{exp(\beta_0 + \beta_1 X_i)}{(1 + exp(\beta_0 + \beta_1 X_i))}$
  3. Complementary Log-Log Model:

$$E(Y_i) = \pi_i = 1 - exp(-exp(\beta_0 + \beta_1 X_i))$$

```
# 1. Probit Mean Response Function

curve(pnorm(0+1*x),-3,3,lwd=2)
curve(pnorm(0+2*x),-3,3,lwd=2,col="red",add=T)

curve(dnorm(x,0,1),-3,3)
curve(exp(x)/(1+exp(x))^2,-3,3,add=T,col="red")
curve(dnorm(x,0,sd=pi/sqrt(3)),-3,3,add=T,col="blue")

# Simulation example:
n=1000
x=sample(0:10,n,replace=T)
line=0.8*x-4
pies=pnorm(line)
plot(x,pies)

y=(runif(n)<pies)
proportions=tapply(y,x,mean)
plot(0:10,proportions)

# Using probit link
results.sim <- glm(y ~ x, family=binomial(link="probit"))
coef(results.sim)
curve(pnorm(-3.703+0.7256*x),0,10,add=T)

# Using logit link
results2.sim <- glm(y ~ x, family=binomial(link="logit"))
coef(results2.im)

plot(0:10,proportions)
curve(exp(-6.77+1.322*x)/(1+exp(-6.77+1.322*x)),0,10,add=T)
curve(pnorm(-3.703+0.7256*x),0,10,add=T,col="darkred")
```

- **Task Example (p. 565).** A systems analyst studied the effect of computer programming experience on ability to complete within a specified time a complex programming task, including debugging. Twenty-five persons were selected for the study. They had varying amounts of programming experience (measured in months). All persons were given the same programming task, and the results of their success ($Y = 1$ if successful and $Y = 0$ if not) in the task are recorded in the file 'Task.csv'.

```
# Task Example (on page 565)
data=read.csv("Task.csv",header=T)
attach(data)
x=experience
y=success

results <- glm(success ~ experience, family=binomial(link="probit"))
results <- glm(success ~ experience, family=binomial(link="logit"))
coef(results)
summary(results)$coef

predicted=exp(-3.0597+0.1615*x)/(1+exp(-3.0597+0.1615*x))
```

Interpretation of β_1: *When x increases by 1 unit, the odds=$\frac{\pi}{(1-\pi)}$ will change by a factor of exp(b_1)=exp(.1615)=1.175272. In other words, it will increase by 17.5% .*

- **Prediction and Errors**

```
prediction=as.numeric(results$fitted>.7) # Using 0.7 as cut off
error=success-prediction
data.frame(success,predicted=results$fitted,prediction,error)

percent.error=sum(abs(error))/length(error)
percent.error

false.pos=sum((success==0)*(prediction==1))
false.neg=sum((success==1)*(prediction==0))

# Finding the best cut off value
cutoff=seq(0,1,by=.05)
m=length(cutoff)
False.pos=array(99,m)
False.neg=array(99,m)
Errors=array(99,m)

for(i in 1:m)
{
prediction=as.numeric(results$fitted>cutoff[i])
error=success-prediction
False.pos[i]=sum((success==0)*(prediction==1))
False.neg[i]=sum((success==1)*(prediction==0))
Errors[i]=False.pos[i]+False.neg[i]
}
Percent.error=Errors/length(Errors)
data.frame(CutOff=cutoff,FalsePositive=False.pos,FalseNegative=False.neg,ErrorRate=Percent.error)
```

- **Hosmer-Lemeshow Goodness of Fit Test**

```
library(ResourceSelection)
hoslem.test(success,results$fitted)

Hosmer and Lemeshow goodness of fit (GOF) test

data:  success, results$fitted
X-squared = 6.56, df = 8, p-value = 0.5848  <-- No evidence to reject the model.
```

- Multiple Logistic Regression

```
dystrophy=read.csv("Dystrophy.csv",header=T)
attach(dystrophy)

dystrophy[!complete.cases(dystrophy),]   # lists rows with missing values
dys.naomit=na.omit(dystrophy)   # creates new data set without missing data
temp=which(is.na(PK)=="TRUE")

results.dys <- glm(carrier ~ AGE+M+CK+H+PK+LD, data = dystrophy, family =binomial(link="logit"))
results.dys.naomit <- glm(carrier ~ AGE+M+CK+H+PK+LD, data = dys.naomit, family =binomial(link="logit"))
summary(results.dys)

carrier.naomit=dys.naomit[,10]
hoslem.test(carrier.naomit,results.dys$fit)
```