## Il problema degli «Sparse Data».

### *Variabili X categoriali*

Consideriamo la seguente tabella di successi (1) e insuccessi (0) per due gruppi di soggetti (A-B):

```
 y<-c(rep(1,8),rep(0,2),rep(1,0),rep(0,10))
 x<-factor(c(rep("A",10),rep("B",10)))
 table(x,y)
##     y
## x     0  1
##   A  2  8
##   B 10  0
```

Come detto, una delle interpretazione del coefficiente $\exp\{\beta\}$ nel caso di variabili indipendenti categoriali è quella di coincidere con **l'odds ratio** (rapporto tra «quote» o «pronostici» di successo):

$$\exp\{\beta\} = \frac{\frac{P(y_i = 1|x_i = B, \beta_j)}{P(y_i = 0|x_i = B, \beta_j)}}{\frac{P(y_i = 1|x_i = A, \beta_j)}{P(y_i = 0|x_i = A, \beta_j)}}$$

Nel nostro caso, il modello di regressione logistica diventa

$$P(y_i = 1|x_i = B, \beta_j) = \frac{exp\{\alpha + \beta D_i\}}{(1 + exp\{\alpha + \beta D_i\})}$$

dove $D_i$ è una variabile Dummy che assume valore 0 = gruppo "A" e 1 = Gruppo "B".

```
n<-length(y)
# Set A as base level
X<-matrix(c(rep(1,n),ifelse(x=="A",0,1)),ncol=2,byrow=F)
```

```
  X
##       [,1] [,2]
## [1,]    1    0
## [2,]    1    0
## [3,]    1    0
## [4,]    1    0
## [5,]    1    0
## [6,]    1    0
## [7,]    1    0
## [8,]    1    0
## [9,]    1    0
##[10,]    1    0
##[11,]    1    1
##[12,]    1    1
##[13,]    1    1
##[14,]    1    1
##[15,]    1    1
##[16,]    1    1
##[17,]    1    1
##[18,]    1    1
##[19,]    1    1
##[20,]    1    1
```

Per i dati tabulati, trasformati in proporzioni condizionali per riga mediante divisione per 10,

```
 table(x,y)/10
##     y
## x     0   1
##   A 0.2 0.8
##   B 1.0 0.0
```

**Nota bene:**

Celle vuote nelle tabelle di contingenza possono determinare stime infinite dei parametri di regressione logistica.

avremo che:

$$\exp\{\hat{\beta}\} = \frac{\frac{0.0}{1.0}}{\frac{0.8}{0.2}} = \frac{0}{4} = 0$$

e di conseguenza, il coefficiente di regressione logistica da stimare avrà valore infinito, infatti:

$$\hat{\beta} = \ln(0) = -\infty$$

$$\beta = \ln(0) = -\infty$$

L'algoritmo iterativo non si accorge di questo limite e continua a ricercare la stima MLE che non arriverà mai, interrompendosi dopo numerosi cicli per singolarità della matrice Hessiana, il cui determinante tenderà a zero, generando quindi (per inversione) stime di varianza infinite (o comunque molto grandi).

```
j=1
i=0
phit0<-matrix(c(0,0))
FisherScoring<-c() ## data set vuoto per i risultati:
repeat{…}          ## Algoritmo di Fisher Scoring

Errore in solve.default(H) :
il sistema è computazionalmente singolare: numero di
condizione reciproca = 1.60167e-16

FisherScoring
     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
[1,]         1.2    1.376932    1.386268    1.386294    1.386294    1.386294
[2,]        -3.2   -4.512267   -5.565089   -6.580431   -7.585980   -8.588010
     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
[1,]    1.386294    1.386294    1.386294    1.386294    1.386294    1.386294
[2,]   -9.588756  -10.589030  -11.589130  -12.589168  -13.589181  -14.589186
     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
[1,]    1.386294    1.386294    1.386294    1.386294    1.386294    1.386294
[2,]  -15.589188  -16.589189  -17.589189  -18.589189  -19.589189  -20.589189
     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
[1,]    1.386294    1.386294    1.386294    1.386294    1.386294    1.386294
[2,]  -21.589189  -22.589189  -23.589189  -24.589189  -25.589189  -26.589189
     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
[1,]    1.386294    1.386294    1.386294    1.386294    1.386294    1.386294
[2,]  -27.589189  -28.589189  -29.589189  -30.589189  -31.589189  -32.589189
     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
[1,]    1.386294    1.386294    1.386294    1.386294    1.386294    1.386294
[2,]  -33.589189  -34.589189  -35.589189  -36.589189  -37.589189  -38.589189
     Result_iter
[1,]    1.386294   log(0.8/0.2)=1.386294
[2,]  -39.589189   −∞

summary(glm(y~x,family=binomial(link=logit)))

…
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    1.3863     0.7906   1.754   0.0795 .
xB           -21.9524  5606.8353  -0.004   0.9969
…
Number of Fisher Scoring iterations: 19
```

## *Variabili X continue*

**Nota bene:**

Anche in questo caso i due coefficiente $\hat{\alpha}$ e $\hat{\beta}$ tendono a valori infiniti.

L'algoritmo iterativo non si accorge di questo limite e continua a ricercare la stima MLE, interrompendosi dopo numerosi cicli per singolarità della matrice Hessiana.

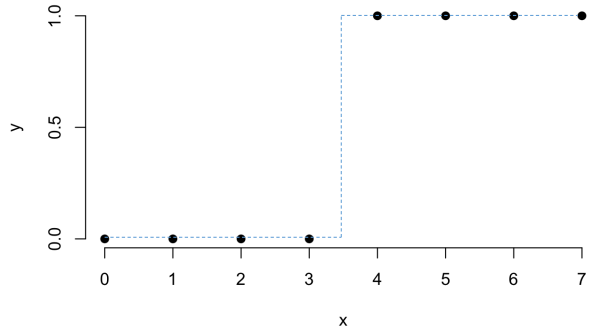
Consideriamo i seguenti dati:

```
x = c(0,1,2,3,4,5,6,7)
y = c(0,0,0,0,1,1,1,1)
n<-length(y)
X<-matrix(c(rep(1,n),x),ncol=2,byrow=FALSE)

j=1
i=0
phit0<-matrix(c(0,0))
FisherScoring<-c() ## data set vuoto per i risultati:
repeat{…}          ## Algoritmo di Fisher Scoring
Errore in solve.default(H) :
Routine Lapack dgesv: il sistema è esattamente singolare:
U[2,2] = 0

FisherScoring
##     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
##[1,]  -2.6666667   -4.700852   -7.381965  -11.485617  -17.730044  -24.955915
##[2,]   0.7619048    1.343101    2.109133    3.281605    5.065727    7.130261
##     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
##[1,]  -32.117093  -39.18380  -46.20916  -53.21859  -60.22207  -67.22335
##[2,]    9.176312   11.19537   13.20262   15.20531   17.20631   19.20667
##     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
##[1,]  -74.22383  -81.22400  -88.22406  -95.22409 -102.22410 -109.22410
##[2,]   21.20681   23.20686   25.20688   27.20688   29.20688   31.20689
##     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
##[1,] -116.22410 -123.22410 -130.22410 -137.22410 -144.22410 -151.22410
##[2,]   33.20689   35.20689   37.20689   39.20689   41.20689   43.20689
##     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
##[1,] -158.22410 -165.22410 -172.22410 -179.22410 -186.22410 -193.22410
##[2,]   45.20689   47.20689   49.20689   51.20689   53.20689   55.20689
##     Result_iter Result_iter Result_iter Result_iter Result_iter Result_iter
##[1,] -200.22410 -207.22410 -214.22410 -221.22410 -228.22410 -235.22410
##[2,]   57.20689   59.20689   61.20689   63.20689   65.20689   67.20689
##     Result_iter Result_iter Result_iter Result_iter
##[1,] -242.22410 -249.22410 -256.22410 -263.22410   −∞
##[2,]   69.20689   71.20689   73.20689   75.20689   +∞

summary(glm(y~x,family=binomial(link=logit)))
[…]
##  Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -160.3   285119.4  -0.001        1
##  x              45.8    80643.9   0.001        1
[…]
## Number of Fisher Scoring iterations: 25

## Messaggio di avvertimento:
## glm.fit: si sono verificate probabilità stimate
## numericamente pari a 0 o 1
```

## *Variabili X continue*

Rappresentiamo graficamente i dati:

```
plot(x,y,pch=19,yaxt="n",bty="n")
axis(2,at=c(0,0.5,1))
```




**Nota bene:**

Si hanno stime infinite quando i valori di x per cui y = 1 possono essere **perfettamente separati** dai valori di x per cui y = 0.

Per spiegare il perché di infinite stime nei parametri di regressione logistica nel caso di dati perfettamente separati, proviamo a disegnare le curve di probabilità previste da 4 modelli distinti:

1) Coefficienti della prima iterazione di Fisher scoring:

```
## valori x generati per disegnare la curva
x.sim<-seq(0,7,by=0.001)
p1<-exp(-2.6666667+x.sim*0.7619048)/
                        (1+exp(-2.6666667+ x.sim*0.7619048))
```

2) Coefficienti della quinta iterazione:

```
p5<-exp(-17.730044+x.sim*5.065727)/
                          (1+exp(-17.730044+x.sim*5.065727))
```

3) Coefficienti della 25-esima iterazione:

```
p25<-exp(-165.22410+x.sim*47.20689)/
                         (1+exp(-165.22410+ x.sim*47.20689))
```

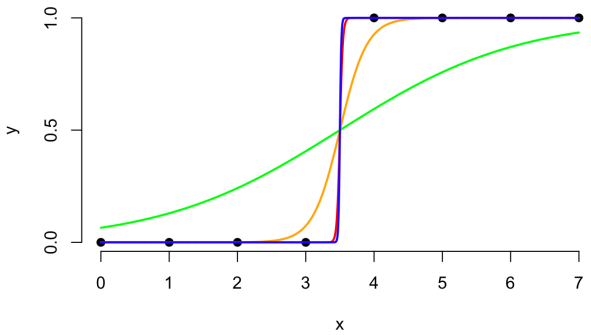
4) Il rapporto $\frac{\hat{\alpha}}{\hat{\beta}} = -3.5$, scegliamo quindi valori arbitrariamente grandi ($\sim\infty$): $\hat{\alpha} = -350$ e $\hat{\beta} = 100$.

```
FisherScoring[1,c(1,5,25)]/FisherScoring[2,c(1,5,25)]
## -3.5        -3.5        -3.5

pInf<-exp(-350+x.sim*100)/
                         (1+exp(-350+x.sim*100))
```

## *Variabili X continue*

Rappresentiamo graficamente le probabilità previste dai quattro modelli:

```
points(x.sim, p1,pch=19,type="l",col="green",lwd=2)
points(x.sim, p5,pch=19,type="l",col="orange",lwd=2)
points(x.sim, p25,pch=19,type="l",col="red",lwd=2)
points(x.sim, pInf,pch=19,type="l",col="blue",lwd=2)
```




**Nota bene:**

Più alto è il valore (in termini assoluti) di $\hat{\alpha}$ e $\hat{\beta}$, più la curva logistica si avvicina ai punti dei dati.

La Residual Deviance tenderà a 0.

**Ricapitolando:**

La stima dei parametri può essere infinita!

Cause:

- *celle vuote nelle tabelle di contingenza (x categoriali),
- * perfetta separazione lungo x tra y=0 e y=1

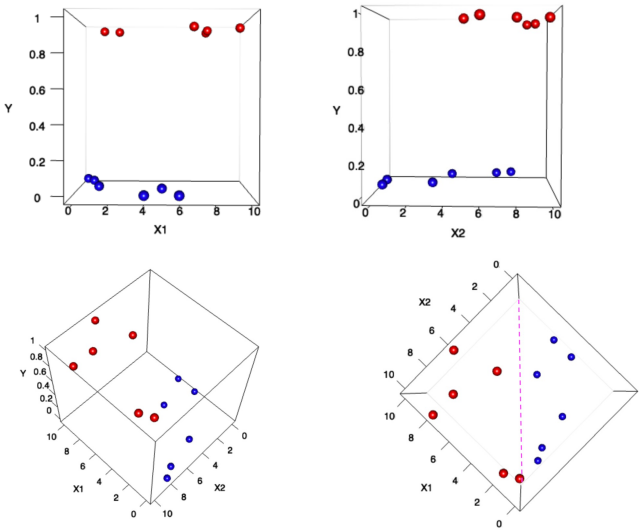
La procedura iterativa può tuttavia convergere ad una stima finita (soprattutto se esistono «blocchi» contro la singolarità della matrice H)

Warnings:

- *Numerosi (*>20/25*) cicli di iterazione Fisher Scoring,
- * Errori standard elevati (*in migliaia*) per i coefficienti stimati

## E il nostro compito per casa?

```
x1 = c(0.5, 2, 10, 6, 8, 1, 8, 4, 0, 7, 5, 1.1)
x2 = c(7, 9.5, 6, 0, 8.1, 9, 10, 0, 8, 5, 3, 4)
y = c(0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -208.05 1127014.63       0        1
x1             21.84  141530.02       0        1
x2             23.19  139959.59       0        1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1.6636e+01  on 11  degrees of freedom
Residual deviance: 6.6586e-10  on  9  degrees of freedom
AIC: 6

Number of Fisher Scoring iterations: 25

Warning messages:
1: glm.fit: algorithm did not converge
2: glm.fit: fitted probabilities numerically 0 or 1
occurred
```




Separazione perfetta tra y=1 e y=0 nel piano X1-X2! (file 3dscatter.html)

## Stima di massima verosimiglianza: Inferenza sui parametri stimati

***Elementi essenziali***

• Data una serie di variabili aleatorie indipendenti, $y_1, y_2, \dots, y_n$ identicamente distribuite secondo una legge di probabilità definita nei parametri $\varphi$, e nei valori della covariata $x_1, x_2, \dots, x_n$

$$p(y_i|x_i) = f(y_i; \varphi, x_i)$$

• la funzione di log- Verosimiglianza può essere scritta come

$$l(\varphi; y_i, x_i) = \sum_{i=1}^{n} ln[f(y_i; \varphi, x_i)] .$$

• La stima di massima verosimiglianza (MLE) richiede la massimizzazione di questa funzione, rispetto ai parametri ignoti $\varphi$.

• Il (vettore) gradiente, *score vector*, contiene le derivate parziali di primo ordine della funzione di verosimiglianza:

$$\boldsymbol{u} = \frac{\partial l(\varphi; y_i, x_i)}{\partial \varphi} = \begin{bmatrix} \frac{\partial l(\varphi; y_i, x_i)}{\partial \varphi_1} \\ \vdots \\ \frac{\partial l(\varphi; y_i, x_i)}{\partial \varphi_j} \end{bmatrix}$$

• Le stime di massima verosimiglianza (MLE) soddisfano la condizione:

$$\boldsymbol{u} = \mathbf{0}$$

• La matrice Hessiana, di ordine $j \times j$, contiene le derivate seconde parziali (e cross-parziali) della funzione di verosimiglianza:

$$\mathbf{H} = \begin{bmatrix} \vdots & \\ \cdots & \dfrac{\partial^2 l(\varphi; y_i, x_i)}{\partial \varphi_j \varphi_j} \quad \cdots \\ \vdots & \end{bmatrix}$$

***In campioni sufficientemente grandi***

• La distribuzione delle stime MLE, $\hat{\varphi}$, è approssimativamente normale:

$$\hat{\varphi} \approx N\left(\varphi, -\hat{\mathbf{H}}^{-1}\right), \qquad \textbf{Eq. 57}$$

la matrice di *varianza-covarianza* dei parametri $\hat{\varphi}$ viene ottenuta dalla matrice Hessiana, laddove possibile (*formule risolutive chiuse*) mediante opportuna sostituzione delle stime MLE nelle formule delle derivate seconde, oppure direttamente al termine del processo iterativo.

• **Wald Test**: supponiamo che sia di interesse eseguire un'inferenza statistica per un singolo parametro $\varphi_j$

• La teoria generale (per grandi campioni) afferma che

$$\hat{\varphi}_j \approx N\left(\varphi_j, -\left[\hat{\mathbf{H}}^{-1}\right]_{jj}\right) \qquad \textbf{Eq. 58}$$

• L'ipotesi nulla circa il valore teorico del coefficiente, $H_0: \varphi_j = 0$, può essere falsificata mediante la statistica di test:

$$\frac{\hat{\varphi}_j - 0}{\sqrt{-\left[\hat{\mathbf{H}}^{-1}\right]_{jj}}} \approx N(0,1) \qquad \textbf{Eq. 59}$$

o in modo equivalente

$$\frac{\left(\hat{\varphi}_j - 0\right)^2}{-\left[\hat{\mathbf{H}}^{-1}\right]_{jj}} \approx z^2 \approx \chi^2_{(1)} \qquad \textbf{Eq. 60}$$

Per il nostro esempio in R:

```
x<-c(8,14,-7,6,5,6,-5,1,0,-17)
y<-c(1,1,0,0,1,0,1,0,0,0)
n<-length(y)
X<-matrix(c(rep(1,n),x),ncol=2,byrow=FALSE)

summary(glm(y~x,family=binomial(link=logit)))

[…]
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.7228     0.8133  -0.889    0.374
x             0.1396     0.1155   1.209    0.227
[…]

###### Fisher Scoring
j=1
i=0
phit0<-matrix(c(0,0))
n<-length(y)
FisherScoring<-c()
###### Applicazione dell'algoritmo iterativo…

               0          1          2
b 0 -0.50923967 -0.6860552 -0.7215888
b 1  0.09930879  0.1328327  0.1394152
               3          4
b 0 -0.7227523 -0.7227534
b 1  0.1396279  0.1396281
```

(…continua)

```
H
##            [,1]       [,2]
## [1,] -1.942224  -6.440556
## [2,] -6.440556 -96.355855
## ---------------------------------

## Test per B0 (j=1)
var_B0 = -1*solve(H)[1,1]
sqrt(var_B0)
##[1] 0.813323
z_B0=(-0.7227534-0)/sqrt(var_B0)
z_B0
##[1] -0.8886425

## Test bilaterale
pnorm(abs(z_B0),lower.tail=FALSE)*2
##[1] 0.3741952

## ---------------------------------
# Test per B1 (j=2)
var_B1 = -1*solve(H)[2,2]
sqrt(var_B1)
##[1] 0.1154712
z_B1=(0.1396 -0)/sqrt(var_B1)
z_B1
##[1] 1.208959

## Test bilaterale
pnorm(abs(z_B1),lower.tail=FALSE)*2
##[1] 0.2266784

## ---------------------------------
## Intervalli di fiducia al 95% per i due coefficienti MLE
-0.7228 + c(-1,+1)*1.96*sqrt(var_B0)
## [1] -2.3169131  0.8713131
0.1396 + c(-1,+1)*1.96*sqrt(var_B1)
## [1] -0.08672356  0.36592356

## Funzione confint()
confint.default(glm(y~x,family=binomial(link=logit)))
##                    2.5 %    97.5 %
## (Intercept) -2.31678189 0.871275
## x           -0.08667973 0.365936

confint(glm(y~x,family=binomial(link=logit)))
Waiting for profiling to be done...
##                    2.5 %    97.5 %
## (Intercept) -2.74573692 0.7181783
## x           -0.04082242 0.4524020
```

• L'ipotesi nulla circa la **combinazione lineare degli elementi in $\varphi$**, ad esempio

$$H_0: \varphi_2 - \varphi_3 = \mathbf{v}'\boldsymbol{\varphi} = 0,$$

con

$$\mathbf{v} = \begin{bmatrix} 0 \\ 1 \\ -1 \\ \vdots \\ 0 \end{bmatrix}$$

e

$$\mathbf{v}'\boldsymbol{\varphi} = [0 \quad 1 \quad -1 \cdots 0] \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \\ \vdots \\ \varphi_j \end{bmatrix} = 0 + \varphi_2 - \varphi_3 + \cdots 0 \times \varphi_j$$

```
x1<-c(8,14,-7,6,5,6,-5,1,0,-17)
x2<-c(2,14,6,4,4,4,-7,2,10,-19)
y<-c(1,1,0,0,1,0,1,0,0,0)
mod<-glm(y~x1+x2,family=binomial(link=logit))
phi<-coefficients(mod)

phi[2]-phi[3]
## 0.8603035

v<-c(0,1,-1) # vettore di contrasti
v%*%phi
##           [,1]
## [1,] 0.8603035
## -Ĥ^-1 matrice di varianza-covarianza dei parametri MLE
n_inv_H<-vcov(mod)
n_inv_H
##              (Intercept)         x1         x2
## (Intercept)   1.1188729 -0.3103417  0.2725780
## x1           -0.3103417  0.2109619 -0.1969017
## x2            0.2725780 -0.1969017  0.1930004
```

secondo la teoria generale per grandi campioni (***Delta method***), si distribuisce come

$$\mathbf{v}'\hat{\boldsymbol{\varphi}} \approx N\left(\mathbf{v}'\boldsymbol{\varphi}, \mathbf{v}'(-\hat{\mathbf{H}}^{-1})\mathbf{v}\right), \qquad \textbf{Eq. 61}$$

può essere falsificata mediante la statistica test:

$$\frac{(\hat{\varphi}_2 - \hat{\varphi}_3) - 0}{\sqrt{\mathbf{v}'(-\widehat{\mathbf{H}}^{-1})\mathbf{v}}} \approx N(0,1) \qquad \textbf{Eq. 62}$$

o in modo equivalente (*Delta Method*):

$$(\mathbf{v}'\hat{\boldsymbol{\varphi}} - \mathbf{0})'\left[\mathbf{v}'(-\widehat{\mathbf{H}}^{-1})\mathbf{v}\right]^{-1}(\mathbf{v}'\hat{\boldsymbol{\varphi}} - \mathbf{0}) \approx \chi^2_{(1)} \qquad \textbf{Eq. 63}$$

```
## Valore della statistica di test, con 1 g.d.l.

Chisq<-t(v%*%phi)%*%solve(t(v)%*%n_inv_H%*%v)%*%(v%*%phi)
Chisq
##           [,1]
##[1,] 0.9277438

1-pchisq(Chisq,df=1) # area della coda Chi-quadrato destra
##           [,1]
## [1,] 0.3354499

## install.packages("car")
library(car) # Usiamo la funzione per contrasti lineari
linearHypothesis(mod, "x1 = x2")
## o in modo equivalente, usando il vettore "v"...
linearHypothesis(mod, v)

## Linear hypothesis test:
## x1 - x2 = 0
##
## Model 1: restricted model
## Model 2: y ~ x1 + x2
##
##   Res.Df Df  Chisq Pr(>Chisq)
## 1      8
## 2      7  1 0.9277     0.3354
```

## *Delta method: Approfondimento*

• Si tratta di un metodo statistico generale per ricavare la varianza di una funzione, $h$, di variabili casuali asintoticamente normali con varianza nota (…*come i coefficienti stimati da un modello LOGIT o di regressione lineare*). Il metodo Delta sfrutta il fatto che $h(\hat{\varphi})$ è (in genere) una funzione differenziabile dei dati e dei parametri del modello, per ricavare una soluzione in forma chiusa per gli errori standard. In particolare,

$$\hat{V}[h(\hat{\varphi})] = \left[\frac{\partial h(\hat{\varphi})}{\partial \hat{\varphi}}\right]^T \left(-\widehat{\mathbf{H}}^{-1}\right)\left[\frac{\partial h(\hat{\varphi})}{\partial \hat{\varphi}}\right]. \qquad \textbf{Eq. 64}$$

• «Jacobiano» è il nome che si usa per il vettore (o matrice) di tutte le derivate parziali di primo ordine di una funzione.
• In breve:
(1) calcoleremo il Jacobiano della funzione $h(\hat{\varphi})$, $\boldsymbol{J}$;
(2) otterremo la matrice di varianza-covarianza, $-\widehat{\mathbf{H}}^{-1}$, dall'output della regressione;
(3) moltiplicheremo *a sandwich*: $\boldsymbol{J}^{\mathrm{T}}\left(-\widehat{\mathbf{H}}^{-1}\right)\boldsymbol{J}$, ottenendo così il nostro valore $\hat{V}[h(\hat{\varphi})]$.

• **Esempio:** Stimiamo la varianza per il **rapporto di due coefficienti di regressione logistica**.

```
 mod$coefficients
## (Intercept)          x1          x2
##  -0.8811331   0.4801557  -0.3801478

Ratio<-0.4801557/(-0.3801478)
 Ratio
## [1] -1.263076
```

## *Delta method: Approfondimento*

1) Calcolare il Jacobiano della funzione $h(\hat{\varphi}) = \hat{\varphi}_2/\hat{\varphi}_3$:

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial h(\hat{\varphi})}{\partial \hat{\varphi}_1} \\ \frac{\partial h(\hat{\varphi})}{\partial \hat{\varphi}_2} \\ \frac{\partial h(\hat{\varphi})}{\partial \hat{\varphi}_3} \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{1}{\hat{\varphi}_2} \\ -\frac{\hat{\varphi}_2}{\hat{\varphi}_3^2} \end{bmatrix}$$

```
   h=expression(phi2/(phi3))
   D(h,"phi2")
## 1/phi3
   D(h,"phi3")
## -(phi2/phi3^2)
   phi<-coefficients(mod)
   J=matrix(c(0,1/phi[3],-phi[2]/phi[3]^2))
   J
##             [,1]
## [1,]  0.000000
## [2,] -2.630556
## [3,] -3.322593
```

2) Ottenere la matrice di varianza-covarianza, $-\widehat{\mathbf{H}}^{-1}$, dall'output della regressione logistica;

```
   vcov(mod)
```

3) Moltiplicazione *a sandwich* $\hat{V}[h(\hat{\varphi})] = \boldsymbol{J}^{\mathrm{T}}\left(-\widehat{\mathbf{H}}^{-1}\right)\boldsymbol{J}$:

```
   t(J)%*%vcov(mod)%*%J
##             [,1]
## [1,] 0.1485244    # Varianza stimata del rapporto

   sqrt(0.1485244)
## [1] 0.3853886     # Errore standard stimato del rapporto


   car::deltaMethod(mod,"x1/x2")
##       Estimate        SE    2.5 %   97.5 %
## x1/x2 -1.26308  0.38539 -2.01842 -0.5077
```

• **Intervallo di fiducia per i valori previsti (probabilità)**:
iniziamo dal Logit, per un generico valore $x_i$

$$\ln\left(\frac{p}{1-p}\right) = \alpha + \beta x_i.$$

In termini matriciali, dato il vettore

$$\mathbf{v} = \begin{bmatrix} 1 \\ x_i \end{bmatrix},$$

$$\mathbf{v}'\boldsymbol{\varphi} = [1 \quad x_i]\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \alpha + \beta x_i$$

con distribuzione, secondo quanto già visto in precedenza,

$$\mathbf{v}'\hat{\boldsymbol{\varphi}} \approx N\left(\mathbf{v}'\boldsymbol{\varphi}, \mathbf{v}'(-\widehat{\mathbf{H}}^{-1})\mathbf{v}\right),$$

e intervallo di fiducia $\mathrm{P}(1-\alpha)$

$$\mathbf{v}'\hat{\boldsymbol{\varphi}} \pm \boldsymbol{z}_{\alpha/2}\sqrt{\mathbf{v}'(-\widehat{\mathbf{H}}^{-1})\mathbf{v}}, \qquad \textbf{Eq. 65}$$

**Nota bene:**

Nel caso di semplici combinazioni lineari di parametri, il vettore Jacobiano si riduce al vettore ***v*** di contrasti:

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial(\alpha+\beta x_i)}{\partial \alpha} \\ \frac{\partial(\alpha+\beta x_i)}{\partial \beta} \end{bmatrix} = \begin{bmatrix} 1 \\ x_i \end{bmatrix} = \mathbf{v}$$

Riprendendo i dati (bivariati) visti in precedenza:

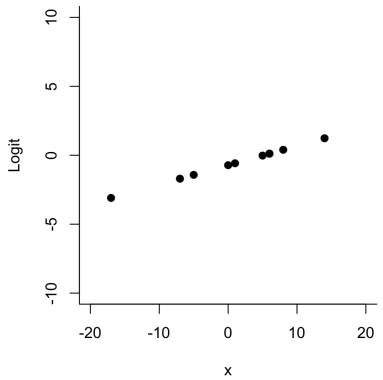
```
x<-c(8,14,-7,6,5,6,-5,1,0,-17)
y<-c(1,1,0,0,1,0,1,0,0,0)
X<-matrix(c(rep(1,n),x),ncol=2,byrow=FALSE)
mod<-glm(y~x,family=binomial(link=logit))
```

per $x_i = 10$

```
v=c(1,10)
phi=coefficients(mod)
t(v)%*%phi
##             [,1]
## [1,] 0.6735278      # Valore Logit previsto per x=10
```

per tutti i valori $x_i$

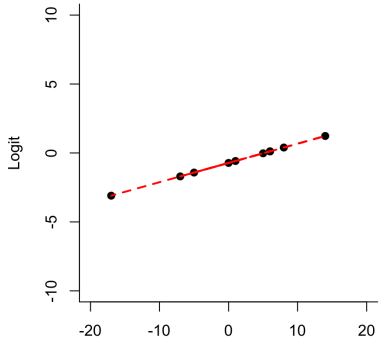
```
# Logit previsti per tutti i valori x
Logit<-X%*%phi
# Plot: valori previsti su scala lineare Logit:
plot(x,Logit,pch=19,bty="l",xlim=c(-20,20),ylim=c(-10,10))
```




Disegniamo il modello teorico lineare

$$\ln\left(\frac{p}{1-p}\right) = \alpha + \beta x_i.$$

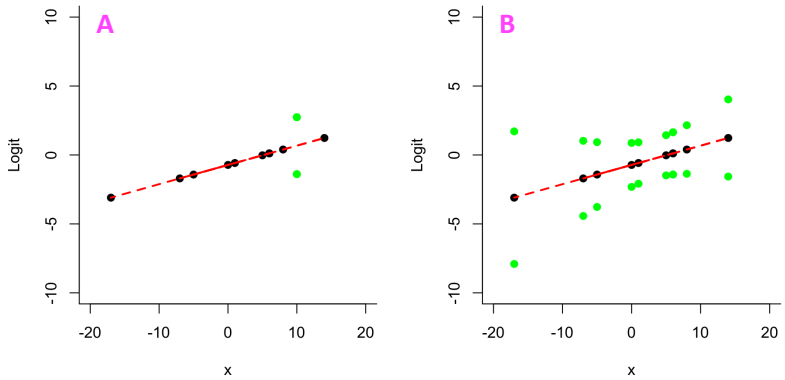
```
# modello lineare teorico
points(x,Logit,pch=19,type="l",col="red",lty="dashed",lwd=2)
```




```
 vcov(mod)
##              (Intercept)           x
## (Intercept)  0.66144838 -0.04420821
## x           -0.04420821  0.01333224
```

A Intervallo di fiducia per il Logit di $x_i = 10$ (Eq. 65) :

```
# Limiti di fiducia al 95% per i valori lineari Logit = a + bx
# inf 95%
Inf95<-(v)%*%phi + 1.96*sqrt(t(v)%*%vcov(mod)%*%v)
##           [,1]
## [1,] 2.738988
# sup 95%
Sup95<-t(v)%*%phi - 1.96*sqrt(t(v)%*%vcov(mod)%*%v)
##            [,1]
## [1,] -1.391933

# disegniamo i due punti estremi:
points(x=c(10,10),y=c(Inf95,Sup95),pch=19, col="green")
```




B Per tutti i valori $x_i$ :

```
Var_Logit<-c()
for(i in 1:length(x)){
              v<-c(1,x[i])
              Var_Logit <- c(Var_Logit, t(v)%*%vcov(mod)%*%v)
                        }
Inf95.logit<- Logit -1.96*sqrt(Var_Logit)
Sup95.logit<- Logit +1.96*sqrt(Var_Logit)

points(x, Inf95.logit,pch=19,col="green",lty="dotted")
points(x, Sup95.logit,pch=19,col="green",lty="dotted")
```

- In termini estesi, notiamo che

$$\hat{V}(\alpha + \beta x_i) = \sqrt{\mathbf{v}'(-\widehat{\mathbf{H}}^{-1})\mathbf{v}}$$

$$= \sqrt{\begin{bmatrix}1 & x_i\end{bmatrix}\begin{bmatrix}Var(\alpha) & Cov(\alpha,\beta)\\ Cov(\alpha,\beta) & Var(\beta)\end{bmatrix}\begin{bmatrix}1\\ x_i\end{bmatrix}}$$

$$= \sqrt{Var(\alpha) + x_i Cov(\alpha,\beta) + x_i Cov(\alpha,\beta) + x_i x_i Var(\beta)}$$

$$= \sqrt{Var(\alpha) + x_i^2 Var(\beta) + 2x_i Cov(\alpha,\beta)}$$

$$= \sqrt{\left[-\widehat{\mathbf{H}}^{-1}\right]_{11} + x_i^2\left[-\widehat{\mathbf{H}}^{-1}\right]_{22} + 2x_i\left[-\widehat{\mathbf{H}}^{-1}\right]_{12}} \quad \textbf{Eq. 66}$$

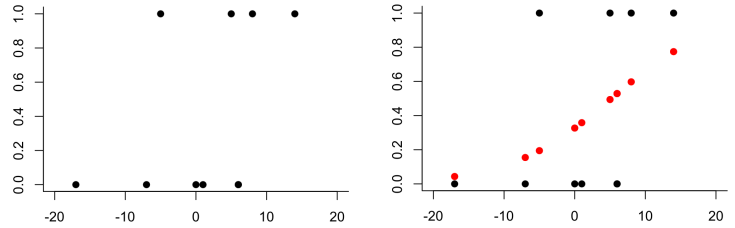
Come previsto dalla regola della varianza di una somma di variabili aleatorie (*vedi Appendice alla lezione*).

```
## Equazione 66
vcov(mod)[1,1] + x^2*vcov(mod)[2,2] + 2*x*vcov(mod)[1,2]

## [1] 0.8073803 2.0367373 1.9336430 0.6109105 0.5526723
## [6] 0.6109105 1.4368364 0.5863642 0.6614484 6.0175444

## calcolato in precedenza
Var_Logit

## [1] 0.8073803 2.0367373 1.9336430 0.6109105 0.5526723
## [6] 0.6109105 1.4368364 0.5863642 0.6614484 6.0175444

## Formula matriciale metodo Delta
diag(X%*%vcov(mod)%*%t(X))

## [1] 0.8073803 2.0367373 1.9336430 0.6109105 0.5526723
## [6] 0.6109105 1.4368364 0.5863642 0.6614484 6.0175444
```

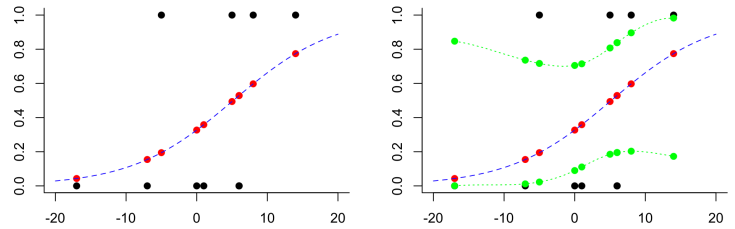
L'intervallo di fiducia per la probabilità di successo si otterrà facilmente, sostituendo i due limiti esterni calcolati per il Logit all'interno della sua funzione inversa (antilogit, *vedi Appendice*):

$$\text{(Logit)}\quad \ln\left(\frac{p}{1-p}\right) = \phi \quad \Rightarrow \quad p = \frac{exp\{\phi\}}{(1 + exp\{\phi\})} \quad \text{(Antilogit)}$$

```
## Antilogit: dal Logit alla probabilità prevista
p<-exp(Logit)/(1+exp(Logit))
plot(x,y,pch=19,bty="l",xlim=c(-20,20),ylim=c(0,1))
points(x,p,pch=19,col="red")
```




```
## Costruiamo la logistica teorica
logistic<-function(x){exp(phi[1]+phi[2]*x)/(1+exp(phi[1]+phi[2]*x))}
curve(logistic,from=-20,to=20,col="blue",lty="dashed",add=TRUE)
```




```
## Limiti di fiducia al 95% per i valori non lineari "p"
## inf 95%
Inf95.antilogit<- exp(Inf95.logit)/(1+exp(Inf95.logit))
## sup 95%
Sup95.antilogit<- exp(Sup95.logit)/(1+exp(Sup95.logit))

points(x, Inf95.antilogit,pch=19,col="green",lty="dotted")
points(x, Sup95.antilogit,pch=19,col="green",lty="dotted")
## Interpolazione continua
points(spline(x, Inf95.antilogit),type="l",col="green",lty="dotted")
points(spline(x, Sup95.antilogit),type="l",col="green",lty="dotted")
```

• **Likelihood ratio test.** Supponiamo che ci interessi verificare simultaneamente l'uguaglianza a zero di un sottoinsieme di parametri:

$$\boldsymbol{H_0}: \varphi_2 = \varphi_3 = 0,$$

```
# dati:
x1<-c(8,14,-7,6,5,6,-5,1,0,-17)
x2<-c(2,14,6,4,4,4,-7,2,10,-19)
y<-c(1,1,0,0,1,0,1,0,0,0)
X<- matrix(c(rep(1,length(y)),x1,x2),byrow=FALSE,ncol=3)

# modello con solo l'intercetta
  mod.H0<-glm(y~1,binomial(link = "logit"))

  summary(mod.H0)

# Call:
# glm(formula = y ~ 1, family = binomial(link = "logit"))
#
# Coefficients:
#             Estimate Std. Error z value Pr(>|z|)
# (Intercept)  -0.4055     0.6455  -0.628     0.53
```

$$\boldsymbol{H_1}: \varphi_2 \neq \varphi_3 \neq 0,$$

```
# modello con i parametri delle covariate diversi da 0
  mod.H1<-glm(y~X[,2]+X[,3],binomial(link = "logit"))

  phi<-coefficients(mod.H1)

  summary(mod.H1)

# Call:
# glm(formula = y ~ X[, 2] + X[, 3], family =
# binomial(link = "logit"))
#
# Coefficients:
#             Estimate Std. Error z value Pr(>|z|)
# (Intercept)  -0.8811     1.0578  -0.833    0.405
# X[, 2]        0.4802     0.4593   1.045    0.296
# X[, 3]       -0.3801     0.4393  -0.865    0.387
```

Si chiama **Deviance** la statistica ottenuta moltiplicando -2 volte il massimo della funzione di log-Verosimiglianza,

```
  Deviance.H1<- -2*(sum(y*X%*%phi-log(1+exp(X%*%phi))))
  Deviance.H1
# [1] 9.344316
  mod.H1$deviance
# [1] 9.344316
```

e distribuita secondo un Chi-quadrato, con gradi di libertà pari alla grandezza campionaria meno il numero di parametri da stimare nel modello lineare.

```
  n=10 # campione = length(y)
  k=2 # variabili x
  Df.H1<-n-k-1
  Df.H1
# [1] 7
```

Per il modello «nullo» contenente unicamente il valore di intercetta avremo quindi:

```
 phi_H0<-c(coefficients(mod.H0),0,0)
 phi_H0
# (Intercept)
#  -0.4054651   0.0000000   0.0000000
  Deviance.H0<- -2*(sum(y*X%*%phi_H0-log(1+exp(X%*%phi_H0))))
  Deviance.H0
# [1] 13.46023
  mod.H0$deviance
# [1] 13.46023

  n=10 # campione = length(y)
  k=0 # solo intercetta
  Df.H0<-n-k-1
  Df.H0
# [1] 9

  summary(mod.H1)

# Null deviance: 13.4602  on 9  degrees of freedom
# Residual deviance:  9.3443  on 7  degrees of freedom
# AIC: 15.344
#
# Number of Fisher Scoring iterations: 6
```

La **differenza tra due Deviance**, una per il modello «nullo» (numericamente più grande) ed una per il modello «alternativo» (numericamente più piccola)

$$-2l(\varphi_0; y_i, x_i) - \left(-2l(\varphi_1; y_i, x_i)\right) =$$

$$= 2\left(l(\varphi_1; y_i, x_i) - l(\varphi_0; y_i, x_i)\right) \approx \chi^2_{(df)}$$

dove *df = gradi di libertà della distribuzione Chi-quadrato*, è la differenza di parametri tra il modello alternativo e quello nullo.

```
# Likelihood ratio test (LRT)
# Distribuzione: Chi-quadrato, df = Df_H0 - Df.H1
# Confronto tra (Deviance) modello nullo e alternativo

  LRT<- Deviance.H0 - Deviance.H1
  LRT
# [1] 4.115918
  Df.H0-Df.H1
# [1] 2
  1-pchisq(LRT,df=2)
# [1] 0.1277144

# Confronto tra i due modelli stimati con glm() e anova()
  anova(mod.H0,mod.H1,test="Chisq")
# Analysis of Deviance Table
#
# Model 1: y ~ 1
# Model 2: y ~ X[, 2] + X[, 3]
#   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
# 1         9    13.4602
# 2         7     9.3443  2   4.1159   0.1277
```

L'ipotesi nulla $\boldsymbol{H_0}: \varphi_2 = \varphi_3 = 0$ non può essere rigettata.

I due predittori $x_1$ e $x_2$ non hanno un effetto statisticamente significativo.

## Deviance e bontà dell'adattamento ai dati (_Goodness of Fit)

- Ad un estremo (inferiore) del processo di stima si trova il modello nullo: è il più semplice possibile, ovvero un modello in cui un unico parametro (intercetta) prevede un valore $p(y = 1|X, B)$ comune a tutti gli $y_i$ .

```
mod.H0<-glm(y~1,binomial(link = "logit"))

## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.4055     0.6455  -0.628     0.53
##
##
##     Null deviance: 13.46  on 9  degrees of freedom
## Residual deviance: 13.46  on 9  degrees of freedom
## AIC: 15.46
```

Nota bene: **Minima** verosimiglianza possibile per i dati → Maggiore Deviance
-2*log(Likelihood) = -2*log(0.001194393) = 13.46

- All'altro estremo (superiore) si trova il modello saturo, caratterizzato da un numero di parametri pari a *n* (o al numero di classi/gruppi di dati che condividono lo stesso valore x).

```
mod.sat<-glm(y~factor(1:length(y)),binomial(link = "logit"))

## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               2.457e+01  1.310e+05       0        1
## factor(1:length(y))2     -1.103e-06  1.853e+05       0        1
## factor(1:length(y))3     -4.913e+01  1.853e+05       0        1
## factor(1:length(y))4     -4.913e+01  1.853e+05       0        1
## factor(1:length(y))5     -1.154e-06  1.853e+05       0        1
## factor(1:length(y))6     -4.913e+01  1.853e+05       0        1
## factor(1:length(y))7     -1.105e-06  1.853e+05       0        1
## factor(1:length(y))8     -4.913e+01  1.853e+05       0        1
## factor(1:length(y))9     -4.913e+01  1.853e+05       0        1
## factor(1:length(y))10    -4.913e+01  1.853e+05       0        1
##
##
##     Null deviance: 1.3460e+01  on 9  degrees of freedom
## Residual deviance: 4.2867e-10  on 0  degrees of freedom
## AIC: 20
```

Nota bene: **Massima** verosimiglianza possibile per i dati → Minore Deviance assoluta (approx. 0)
-2*log(Likelihood) = -2*log(1) = 0.00000…

❖ Un modello "buono" si colloca solitamente tra questi due estremi, poiché un modello saturo è inutile in quanto incapace di sintetizzare i dati (si limita infatti a ripeterli integralmente), mentre un modello nullo è solitamente troppo semplice per essere utile. Bisogna cercare un equilibrio tra gli obiettivi parsimonia e bontà di adattamento.

```
mod.H1<-glm(y~X[,2]+X[,3],binomial(link = "logit"))

## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.8811     1.0578  -0.833    0.405
## X[, 2]        0.4802     0.4593   1.045    0.296
## X[, 3]       -0.3801     0.4393  -0.865    0.387
##
##
##     Null deviance: 13.4602  on 9  degrees of freedom
## Residual deviance:  9.3443  on 7  degrees of freedom
## AIC: 15.344
```

Nota bene: verosimiglianza e Deviance intermedie rispetto ai due modelli precedenti.
-2*log(Likelihood) = -2*log(0.009352066) = 9.344316

❖Se la Verosimiglianza del modello è alta [*Limite sup: P=1*], allora la Deviance avrà un valore basso [*Limite Inf.:* $-2\ln(1) = 0$].

❖Se la Verosimiglianza del modello è bassa [*Limite inf: P=0*], allora la Deviance avrà un valore alto [*Limite Sup.:* $-2\ln(0) = +\infty$].

❖La Deviance rappresenta quindi una misura di *"scostamento dai dati"*: se il suo valore è basso allora *l'adattamento è migliore*.

❖Possiamo valutare la funzione di Verosimiglianza per il modello saturo e per il modello attuale (modello H1) ai valore delle stime MLE, ottenuta in entrambi i casi. Se il modello attuale (modello H1) si adatta bene ai dati, le due funzioni di Verosimiglianza dovrebbero avere valori molto simili:

❖ Formalmente, la quantità

$$D = 2\{l(\varphi_{saturo}; y_i, x_i) - l(\varphi_1; y_i, x_i)\}$$

$$= 2\{0 - l(\varphi_1; y_i, x_i)\} = -2l(\varphi_1; y_i, x_i)$$

È chiamata Deviance del modello (*a.k.a., Residual Deviance*)

❖ Questa quantità è piccola per i modelli «buoni» ed è grande quando il modello attuale (H1) fornisce un adattamento scadente.

❖ Il modello saturo è inutile, ma $l(\varphi_{saturo}; y_i, x_i)$ fornisce un punto di riferimento per confrontare la log-Verosimiglianza del modello attuale (H1).

```
logLik(mod.sat)
## 'log Lik.' -2.143334e-10 (df=10) #parametri nel modello!!
                      # Fit perfetto.


logLik(mod.H0)
## 'log Lik.' -6.730117 (df=1)      #parametri nel modello!!
1-pchisq(-2*c(logLik(mod.H0)),df=n-1)
## [1] 0.1428628    # Non significa. diverso dal modello saturo.


logLik(mod.H1)
## 'log Lik.' -4.672158 (df=3)      #parametri nel modello!!
1-pchisq(-2*c(logLik(mod.H1)),df=n-3)
## [1] 0.2288716   # Non significa. diverso dal modello saturo

##LRT
LRT<- (-2*c(logLik(mod.H0))) - (-2*c(logLik(mod.H1)))
LRT
## [1] 4.115918
Df=9-7
1-pchisq(LRT,df=Df)
## [1] 0.1277144
# Modello nullo e modello H1 non differiscono significativamente
# tra loro. Per parsimonia, considereremo i due predittori x1 e x2
# non significativi e quindi rimovibili!
```

**Appendice alla lezione.**

• Consideriamo due variabili aleatorie *X* e *Y*. Definiamo la covarianza tra X e Y, scritta come *Cov (X,Y):*

$$Cov(X,Y) = E[(X - EX)(Y - EY)]$$

$$= E[XY - X(EY) - (EX)Y + (EX)(EY)]$$

$$= E[XY] - E[X(EY)] - E[(EX)Y] + (EX)(EY)$$

$$= E[XY] - (EX)(EY) - (EX)(EY) + (EX)(EY)$$

$$= E[XY] - (EX)(EY)$$

• La varianza di *X* (o *Y*)

$$Var(X) = E[X^2] - (EX)^2,$$

è un caso limite della covarianza (di una variabile con se stessa!):

$$Cov(X,X) = E[(X - EX))(X - EX)]$$

$$= E[XX] - (EX)(EX) - (EX)(EX) + (EX)(EX)$$

$$= E[X^2] - (EX)^2.$$

• Supponendo che $Z = X + Y$, allora avremo che $Var(Z) = Cov(Z,Z)$

$$= Cov(X + Y, X + Y)$$

$$= E[(X + Y)^2] - [E(X + Y)]^2$$

$$= E[X^2 + Y^2 + 2XY] - [EX + EY]^2$$

$$= E[X^2] + E[Y^2] + 2E[XY] - (EX)^2 - (EY)^2 - 2(EX)(EY)$$

$$= \underbrace{E[X^2] - (EX)^2}_{Cov(X,X)} + \underbrace{E[Y^2] - (EY)^2}_{Cov(Y,Y)} + \underbrace{2E[XY] - 2(EX)(EY)}_{2Cov(X,Y)}$$

$$= Var(X) + Var(Y) + 2 * Cov(X,Y)$$

• Questo risultato viene spesso indicato come (proprietà della) **bilinearità** della covarianza:

$$Cov(X + Y, Z + J) = Cov(X,Z) + Cov(X,J) + Cov(Y,Z) + Cov(Y,J)$$

## Appendice alla lezione.

• Nel caso del Logit, per un generico valore $x_i$

$$ln\left[\frac{p(y_i=1|x_i,\beta_j)}{1-p(y_i=1|x_i,\beta_j)}\right] = \alpha + \beta x_i,$$

avremo che $Var\left(ln\left[\frac{p(y_i=1|x_i,\beta_j)}{1-p(y_i=1|x_i,\beta_j)}\right]\right) = Var(\alpha + \beta x_i)$

$$= Cov(\alpha + \beta x_i, \alpha + \beta x_i)$$

$$= Cov(\alpha,\alpha) + Cov(\beta x_i, \beta x_i) + 2Cov(\alpha, \beta x_i)$$

$$= Var(\alpha) + x_i^2 E[\beta^2] - x_i^2 (E\beta)^2 + 2x_i E[\alpha\beta] - 2x_i (E\alpha)(E\beta)$$

$$= Var(\alpha) + x_i^2 Var(\beta) + 2x_i Cov(\alpha,\beta)$$

$$= \left[-\widehat{\mathbf{H}}^{-1}\right]_{11} + x_i^2\left[-\widehat{\mathbf{H}}^{-1}\right]_{22} + 2x_i\left[-\widehat{\mathbf{H}}^{-1}\right]_{12}$$

• **Dal Logit alla probabilità di successo** prevista dal modello di regressione logistica:

$$\ln\left(\frac{p}{1-p}\right) = \phi \qquad \text{(Logit)}$$

$$\left(\frac{p}{1-p}\right) = exp\{\phi\}$$

$$p = exp\{\phi\}(1-p)$$

$$p = exp\{\phi\} - exp\{\phi\}p$$

$$p + exp\{\phi\}p = exp\{\phi\}$$

$$p(1 + exp\{\phi\}) = exp\{\phi\}$$

$$p = exp\{\phi\}/(1 + exp\{\phi\}) \qquad \text{(Antilogit)}$$