Laboratori del corso di Statistica (cp)

2. Regressione Poisson

Leonardo Egidi

AA 2021/2022

Assicurazioni auto

Leggiamo l'archivio `car.csv` (De Jong and Heller, 2008)

```
cars <- read.csv(file="car.csv",header=TRUE,sep=";")
```

Il campione riguarda delle polizze RC auto annuali, in forza nel 2004 o 2005. In particolare si hanno per 67856 polizze le seguenti variabili.

- `exposure` ($\in [0,1]$) porzione dell'anno in cui la polizza è osservata (periodo in cui è esposta al rischio).
- `numclaims` numero di sinistri
- `claimcst0` valore dei sinistri (0 se `numclaims` è 0)
- `veh_value` valore del veicolo in \$10,000
- `veh_body` tipo di veicolo (13 valori)
- `veh_age` vetustà del veicolo: 1 (più recente), 2, 3, 4
- `gender` sesso del guidatore: M, F
- `area` residenza del guidatore: A, B, C, D, E, F
- `agecat` età del guidatore: 1 (più giovane), 2, 3, 4, 5, 6

Possiamo visualizzare la struttura del `data.frame` con

```
head(cars)
```

```
##   veh_value  exposure numclaims claimcst0 veh_body veh_age gender
## 1      1.06 0.3039014         0         0    HBACK       3      F
## 2      1.03 0.6488706         0         0    HBACK       2      F
## 3      3.26 0.5694730         0         0      UTE       2      F
## 4      4.14 0.3175907         0         0    STNWG       2      F
## 5      0.72 0.6488706         0         0    HBACK       4      F
## 6      2.01 0.8542094         0         0    HDTOP       3      M
##   area agecat
## 1    C      2
## 2    A      4
## 3    E      2
## 4    D      2
## 5    C      2
## 6    C      4
```

```
str(cars)
```

```
## 'data.frame': 67856 obs. of  9 variables:
##  $ veh_value: num  1.06 1.03 3.26 4.14 0.72 2.01 1.6 1.47 0.52 0.38 ...
##  $ exposure : num  0.304 0.649 0.569 0.318 0.649 ...
##  $ numclaims: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ claimcst0: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ veh_body : chr  "HBACK" "HBACK" "UTE" "STNWG" ...
##  $ veh_age  : int  3 2 2 2 4 3 3 2 4 4 ...
##  $ gender   : chr  "F" "F" "F" "F" ...
##  $ area     : chr  "C" "A" "E" "D" ...
##  $ agecat   : int  2 4 2 2 2 4 4 6 3 4 ...
```

Notiamo che alcune variabili che hanno natura categoriale sono codificate come numeri, è conveniente ricodificarle

```
cars$veh_age <- as.factor(cars$veh_age)
cars$agecat <- as.factor(cars$agecat)
```

(notare i cambiamenti usando di nuovo il comando `str(cars)`).

Analisi esplorativa

La maggior parte delle variabili che interessa esaminare sono categoriali, per analizzare le distribuzioni marginali si userà un diagramma a barre della distribuzione di frequenze.
Considerando ad esempio `numclaims` si ottiene la tabella di frequenze con

```
a <- table(cars$numclaims)
a

##
##     0     1     2     3     4
## 63232  4333   271    18     2
```

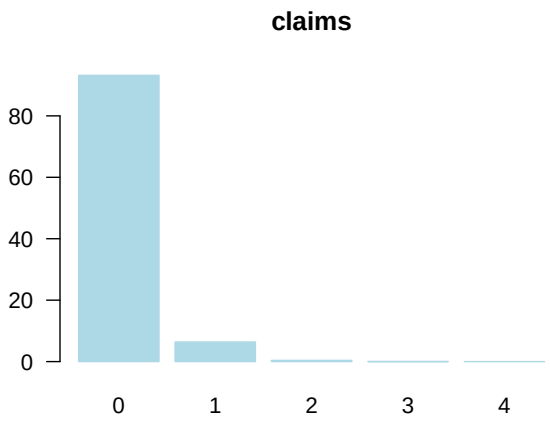
Possiamo anche usare la funzione `prop.table` per ottenere la distribuzione delle frequenze relative

```
a <- prop.table(table(cars$numclaims))
a

##
##             0             1             2             3             4
## 9.318557e-01 6.385581e-02 3.993751e-03 2.652676e-04 2.947418e-05
```

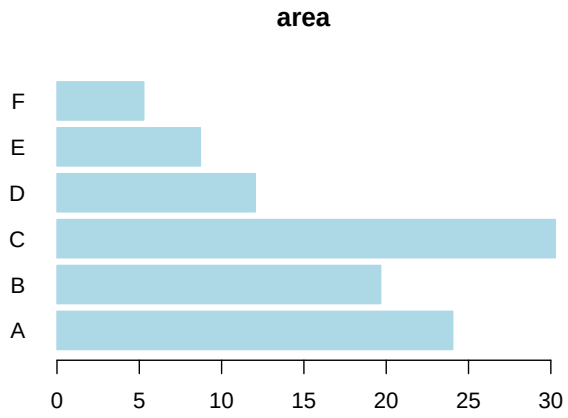
Visualizziamo quest'ultima con un diagramma a barre.

```
barplot(100*a, col="lightblue", border="lightblue", las=1, main="claims")
```

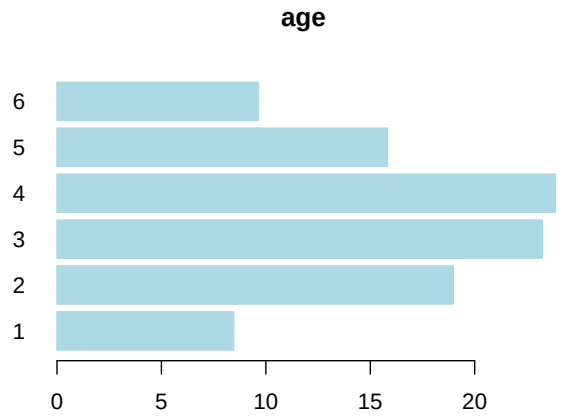


Le istruzioni seguenti visualizzano le distribuzioni di frequenza per tutte le variabili qualitative.

```
a <- prop.table(table(cars$area))
barplot(100*a, col="lightblue", border="lightblue", las=1, horiz=TRUE, main="area")
```

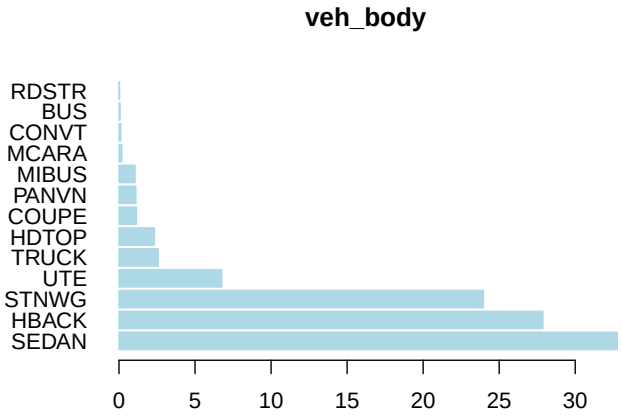



```
a <- prop.table(table(cars$agecat))
barplot(100*a, col="lightblue", border="lightblue", las=1, horiz=TRUE, main="age")
```



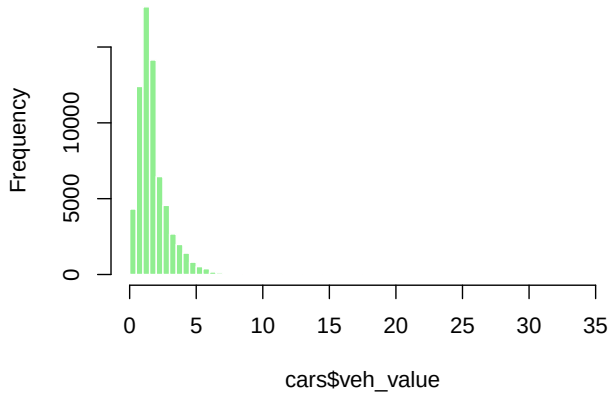
Nel caso di `veh_body` può essere conveniente ordinare i livelli in base alla frequenza per ottenere un grafico più leggibile, inoltre conviene girare di 90 gradi il grafico per lasciare spazio alle etichette.

```
cars$veh_body <- factor(cars$veh_body, levels=names(sort(table(cars$veh_body),
                        decreasing=TRUE)))
a <- prop.table(table(cars$veh_body))
barplot(100*a, col="lightblue", border="lightblue", las=1, horiz=TRUE,
        main="veh_body")
```

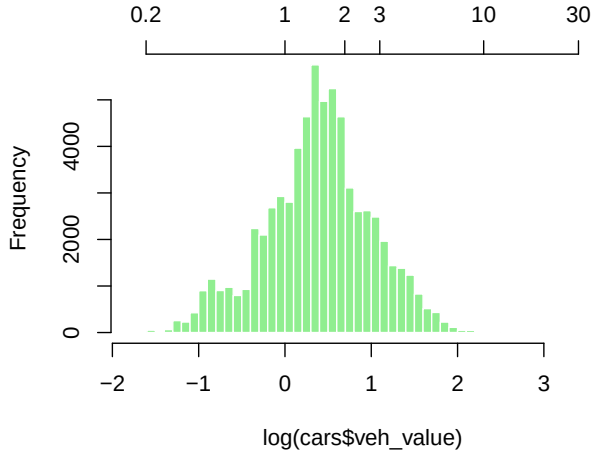


Per quanto riguarda `veh_value`, il grafico originale è poco chiaro a causa dell'asimmmetria della variabile, conviene rappresentare anche il logaritmo.

```
hist(cars$veh_value, col="lightgreen", border="white", main="", br=50)
```



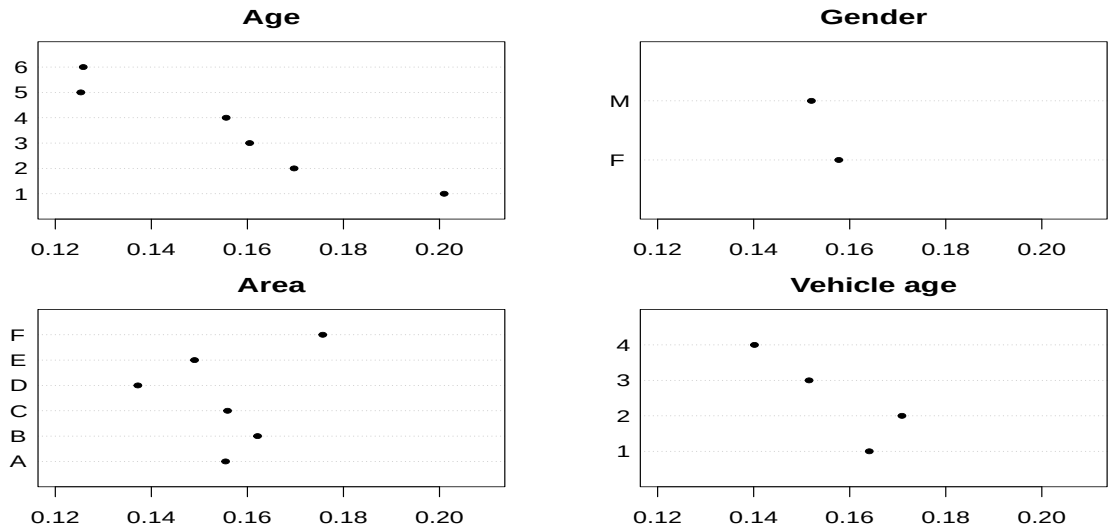

```
hist(log(cars$veh_value), col="lightgreen", border="white", main="", br=50)
y.orig <- c(0.2, 1, 2, 3, 10, 30)
axis(3, at=log(y.orig), labels=y.orig)
```



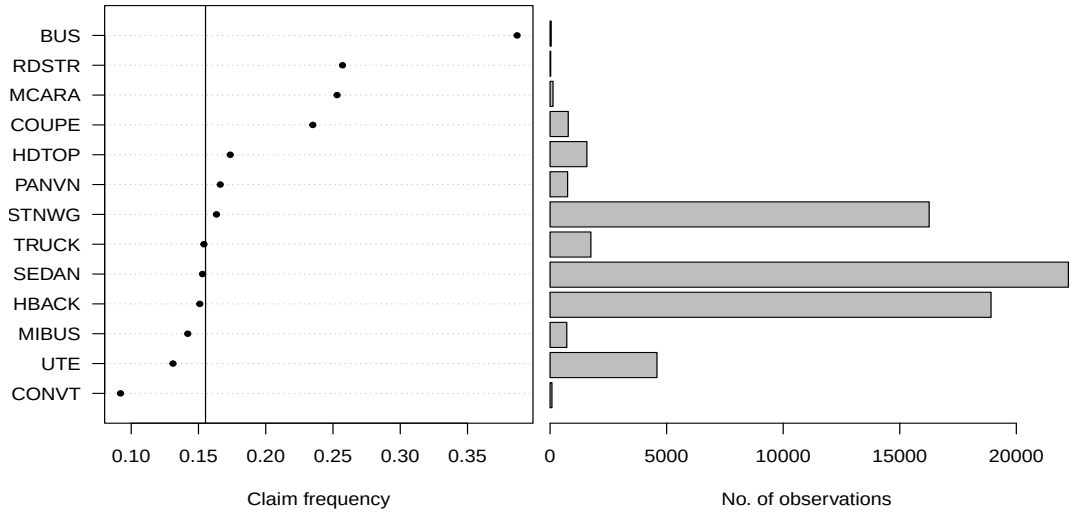
Relazione con numclaim

Investighiamo la relazione tra il numero di sinistri e le esplicative.

```
cars$muta=1
par(mfrow=c(2,2),mar=c(2,4,2.5,0.5),cex=1.5,las=1)
freqBy=with(cars,by(numclaims,agecat,sum))/with(cars,by(exposure,agecat,sum))
dotchart(as.numeric(freqBy),pch=20,main="Age",xlim=c(0.12,0.21),labels=names(freqBy))
freqBy=with(cars,by(numclaims,gender,sum))/with(cars,by(exposure,gender,sum))
dotchart(as.numeric(freqBy),pch=20,main="Gender",xlim=c(0.12,0.21),labels=names(freqBy))
freqBy=with(cars,by(numclaims,area,sum))/with(cars,by(exposure,area,sum))
dotchart(as.numeric(freqBy),pch=20,main="Area",xlim=c(0.12,0.21),labels=names(freqBy))
freqBy=with(cars,by(numclaims,veh_age,sum))/with(cars,by(exposure,veh_age,sum))
dotchart(as.numeric(freqBy),pch=20,main="Vehicle age",xlim=c(0.12,0.21),labels=names(freqBy))
```



```
freqBbody=with(cars,by(numclaims,veh_body,sum))/with(cars,by(exposure,veh_body,sum))
bodyTable=with(cars,table(veh_body))
sl=sort.list(freqBbody)
par(mfrow=c(1,2),mar=c(5,4,0.5,0.5))
dotchart(as.numeric(freqBbody[sl]),pch=20,xlab="Claim frequency")
axis(2,at=1:length(freqBbody),labels=names(freqBbody)[sl],las=1)
abline(v=with(cars,mean(numclaims)/mean(exposure)))
par(mar=c(5,0,0.5,0.5))
barplot(bodyTable[sl],horiz=TRUE,yaxt="n",xlab="No. of observations")
```



Stima della regressione di Poisson

La prima questione di cui tenere conto è che le polizze non sono esposte al rischio per lo stesso tempo, la variabile `exposure` riporta la frazione di anno per cui il veicolo è coperto.
Il numero di sinistri è naturalmente legato alla variabile di esposizione e questa può essere considerata un *offset*, si stima cioè un modello del tipo

$$y_i \sim \text{Poisson}(\lambda_i)$$

$$\log\left(\frac{\lambda_i}{e_i}\right) = \mathbf{x}_i^T \boldsymbol{\beta}$$

Per apprezzare il ruolo dell'*offset* consideriamo dapprima un modello senza esplicative, in cui cioè

$$\log \lambda_i = \beta_0 + \log e_i$$

che stimiamo con

```
fit0=glm(numclaims~1+offset(log(exposure)),family=poisson,data=cars)
summary(fit0)
```

```
##
## Call:
## glm(formula = numclaims ~ 1 + offset(log(exposure)), family = poisson,
##     data = cars)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.5570  -0.4573  -0.3511  -0.2254   4.4368
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.86273    0.01423  -130.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 25507  on 67855  degrees of freedom
## Residual deviance: 25507  on 67855  degrees of freedom
## AIC: 34944
##
## Number of Fisher Scoring iterations: 6
```

otteniamo $\widehat{\beta}_0 = -1.863$, che corrisponde a un tasso annuale di sinistri pari a $e^{\widehat{\beta}_0} = 0.155$.
Si noti che quest'ultimo non è altro che

$$\frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} e_i}$$

```
sum(cars$numclaims)/sum(cars$exposure)

## [1] 0.1552476
```

che è chiaramente maggiore del numero medio di sinistri

$$\frac{\sum_{i=1}^{n} y_i}{n}$$

```
mean(cars$numclaims)

## [1] 0.07275701
```

essendo tutte le esposizioni minori di uno.
Consideriamo ora un modello con l'offset e l'esplicativa `agecat`

```
fitA <- glm(numclaims ~ agecat, family=poisson(link=log),
            offset=log(exposure), data=cars)
summary(fitA)
```

```
##
## Call:
## glm(formula = numclaims ~ agecat, family = poisson(link = log),
##     data = cars, offset = log(exposure))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6338  -0.4544  -0.3482  -0.2227   4.5443
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.60458    0.04364 -36.766  < 2e-16 ***
## agecat2     -0.16900    0.05390  -3.136  0.00172 **
## agecat3     -0.22507    0.05240  -4.295 1.75e-05 ***
## agecat4     -0.25600    0.05243  -4.883 1.05e-06 ***
## agecat5     -0.47235    0.05872  -8.044 8.68e-16 ***
## agecat6     -0.46832    0.06685  -7.006 2.46e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 25507  on 67855  degrees of freedom
## Residual deviance: 25415  on 67850  degrees of freedom
## AIC: 34862
##
## Number of Fisher Scoring iterations: 6
```

Dalla tabella dei coefficienti possiamo concludere che la variabile `agecat` è significativa, e che il tasso annuo di sinistri diminuisce con l'età. L'intercetta è il logaritmo del tasso annuo di sinistri per la categoria 1, e quindi il tasso è

```
exp(coef(fitA)[1])
```

```
## (Intercept)
##   0.2009743
```

Possiamo calcolare il tasso annuo di sinistri per la categoria `j` come

```
exp(coef(fitA)[1]+coef(fitA)[j])
```

calcoliamolo per tutte le categorie come

```r
exp(coef(fitA)[1]+coef(fitA)[-1])
```

```
##   agecat2   agecat3   agecat4   agecat5   agecat6
## 0.1697254 0.1604706 0.1555824 0.1253140 0.1258200
```

Naturalmente, lo stesso si può ottenere con `predict`, specificando `exposure` pari a `1`.

```r
predict(fitA,type="response",
        newdata=data.frame(agecat=factor(1:6),exposure=1))
```

```
##         1         2         3         4         5         6
## 0.2009743 0.1697254 0.1604706 0.1555824 0.1253140 0.1258200
```

Si noti che le stime dei tassi annui non sono altro che i rapporti, per ciascuna categoria tra numero di sinistri e esposizione, che si possono calcolare da

```r
by(cars,cars$agecat,FUN=function(d) sum(d$numclaims)/sum(d$exposure))
```

```
## cars$agecat: 1
## [1] 0.2009743
## ------------------------------------------------------------
## cars$agecat: 2
## [1] 0.1697254
## ------------------------------------------------------------
## cars$agecat: 3
## [1] 0.1604706
## ------------------------------------------------------------
## cars$agecat: 4
## [1] 0.1555824
## ------------------------------------------------------------
## cars$agecat: 5
## [1] 0.125314
## ------------------------------------------------------------
## cars$agecat: 6
## [1] 0.12582
```

Selezione del modello

Dobbiamo poi scegliere quali variabili includere nel modello tra quelle a disposizione, possiamo adottare una procedura a passi. Le variabili candidate sono

`veh_value` `veh_body` `veh_age` `gender` `area` `agecat`

Possiamo usare a tal fine l'istruzione `step`.

```r
fitSTEP <- step(fitA, scope= ~ veh_value + veh_body + veh_age + gender
                              + area + agecat, direction="both")
```

```
## Start:  AIC=34862.03
## numclaims ~ agecat
##
##             Df Deviance   AIC
## + veh_age    3    25388 34841
## + veh_value  1    25396 34845
## + veh_body  12    25376 34847
## + area       5    25404 34860
## <none>            25415 34862
## + gender     1    25415 34863
## - agecat     5    25507 34944
##
## Step:  AIC=34840.93
## numclaims ~ agecat + veh_age
##
##             Df Deviance   AIC
## + veh_body  12    25345 34822
## + veh_value  1    25383 34837
## + area       5    25377 34840
## <none>            25388 34841
## + gender     1    25388 34843
## - veh_age    3    25415 34862
## - agecat     5    25478 34920
##
## Step:  AIC=34821.84
## numclaims ~ agecat + veh_age + veh_body
##
##             Df Deviance   AIC
## + area       5    25334 34821
## <none>            25345 34822
## + veh_value  1    25343 34822
## + gender     1    25345 34823
## - veh_body  12    25388 34841
## - veh_age    3    25376 34847
## - agecat     5    25436 34903
##
## Step:  AIC=34820.98
## numclaims ~ agecat + veh_age + veh_body + area
##
##             Df Deviance   AIC
## <none>            25334 34821
```

```
## + veh_value  1    25333 34821
## - area       5    25345 34822
## + gender     1    25334 34822
## - veh_body  12    25377 34840
## - veh_age    3    25365 34845
## - agecat     5    25422 34898
```

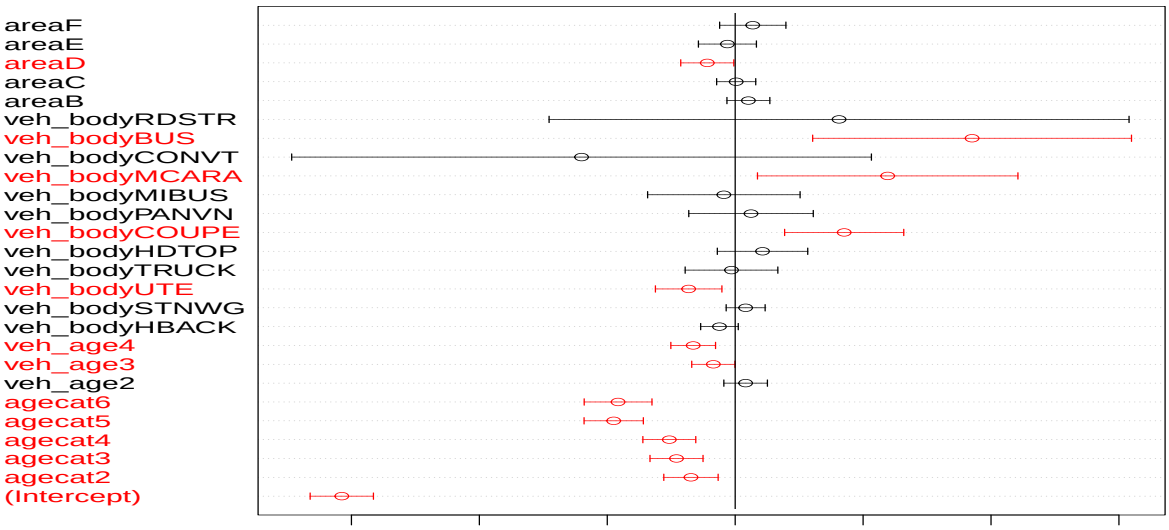
```
summary(fitSTEP)

##
## Call:
## glm(formula = numclaims ~ agecat + veh_age + veh_body + area,
##     family = poisson(link = log), data = cars, offset = log(exposure))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8985  -0.4521  -0.3460  -0.2213   4.5532
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.537677   0.063099 -24.369  < 2e-16 ***
## agecat2       -0.172843   0.054178  -3.190 0.001421 **
## agecat3       -0.229309   0.052889  -4.336 1.45e-05 ***
## agecat4       -0.257461   0.052743  -4.881 1.05e-06 ***
## agecat5       -0.474815   0.059105  -8.033 9.48e-16 ***
## agecat6       -0.457768   0.067583  -6.773 1.26e-11 ***
## veh_age2       0.040850   0.043445   0.940 0.347082
## veh_age3      -0.085385   0.043090  -1.982 0.047529 *
## veh_age4      -0.164123   0.044584  -3.681 0.000232 ***
## veh_bodyHBACK -0.061465   0.037463  -1.641 0.100861
## veh_bodySTNWG  0.040907   0.038849   1.053 0.292363
## veh_bodyUTE   -0.181847   0.066302  -2.743 0.006094 **
## veh_bodyTRUCK -0.014401   0.092422  -0.156 0.876177
## veh_bodyHDTOP  0.106881   0.090149   1.186 0.235776
## veh_bodyCOUPE  0.426092   0.118783   3.587 0.000334 ***
## veh_bodyPANVN  0.061901   0.124132   0.499 0.618014
## veh_bodyMIBUS -0.044306   0.152034  -0.291 0.770730
## veh_bodyMCARA  0.596160   0.259702   2.296 0.021701 *
## veh_bodyCONVT -0.600532   0.578065  -1.039 0.298867
## veh_bodyBUS    0.925994   0.317916   2.913 0.003583 **
## veh_bodyRDSTR  0.406066   0.578314   0.702 0.482582
## areaB          0.051626   0.042778   1.207 0.227496
## areaC          0.003946   0.038977   0.101 0.919371
```

```
## areaD         -0.109084   0.052918  -2.061 0.039267 *
## areaE         -0.030349   0.057846  -0.525 0.599825
## areaF          0.068747   0.066070   1.041 0.298100
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 25507  on 67855  degrees of freedom
## Residual deviance: 25334  on 67830  degrees of freedom
## AIC: 34821
##
## Number of Fisher Scoring iterations: 6
```

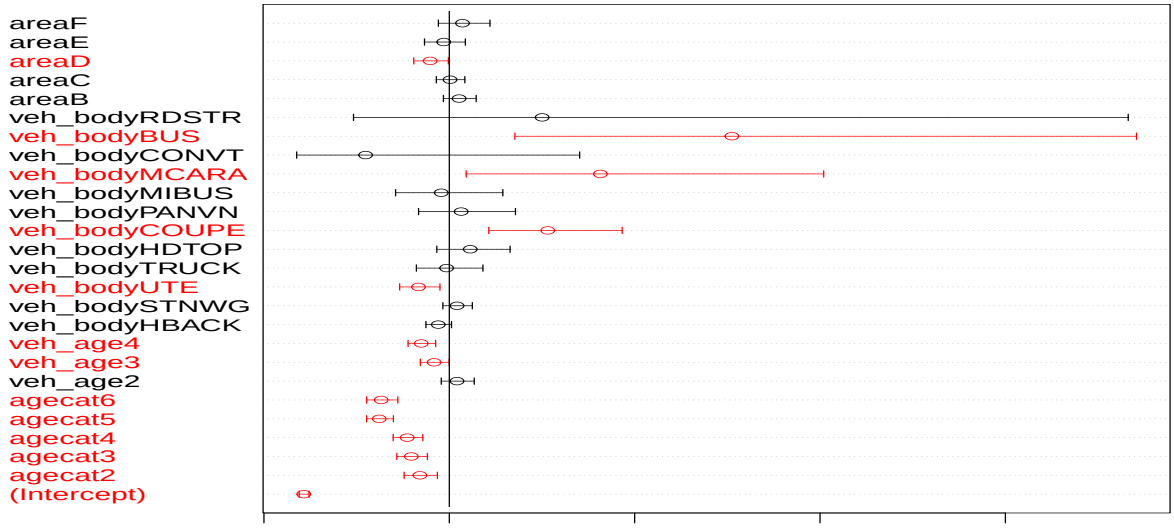
Può essere utile rappresentare i coefficienti graficamente

```
sfitSTEP=summary(fitSTEP)
par(mar=c(1,4,1,1),cex=1.5)
up=sfitSTEP$coef[,1]+1.96*sfitSTEP$coef[,2]
low=sfitSTEP$coef[,1]-1.96*sfitSTEP$coef[,2]
signif=!(0<up & 0>low)
colore=c("black","red")[1+signif]
dotchart(sfitSTEP$coef[,1],xlim=range(low,up),xlab=expression(hat(beta)[i]),col=colore)
arrows(low,1:nrow(sfitSTEP$coef),up, 1:nrow(sfitSTEP$coef), length=0.05, angle=90, code=3,
       col=colore)
abline(v=0)
```



La stessa cosa può essere fatta per $e^{\widehat{\beta}_j}$.

```
par(mar=c(1,4,1,1),cex=1.5)
up2=exp(up)
low2=exp(low)
signif=!(1<up2 & 1>low2)
colore=c("black","red")[1+signif]
dotchart(exp(sfitSTEP$coef[,1]),xlim=range(low2,up2),xlab=expression(e^hat(beta)[i]),
         col=colore)
arrows(low2,1:nrow(sfitSTEP$coef), up2, 1:nrow(sfitSTEP$coef), length=0.05, angle=90,
       code=3,col=colore)
abline(v=1)
```



Si noti che la procedura *stepwise* include o esclude dal modello le variabili qualitative, che corrispondono a un blocco di variabili indicatrici, alcune delle quali possono risultare non significative. Ad esempio si noti che per `area` è significativa solo `areaD`, cioè la variabile indicatrice che è `1` per `area=D` e 0 altrimenti (a significare che c'è una differenza significativa tra area `A`, la *baseline*, e area `D` ma non tra l'area `A` e le altre).

Modellare le unità di terapia intensiva Covid-19

Descrizione del dataset

Ci concentriamo ora sulla modellazione del numero di unità di terapia intensiva (ICU) registrate nelle regioni italiane durante la prima fase dell'epidemia Covid-19. I dati sono disponibili sul sito web della **Protezione Civile** `https://github.com/pcm-dpc/COVID-19` e sono scaricabili direttamente dal sito. Il dataset `covid19_ita.csv` raccoglie 2121 osservazioni: per ogni regione abbiamo una *serie temporale* di 101 osservazioni (non solo per ICU), dal 24 febbraio al 3 giugno 2020.
La costruzione di un modello predittivo per le terapia intensive consente una migliore allocazione dei pazienti durante la diffusione dell'epidemia: naturalmente, ci sono molte covariate che possono

aiutare in questo compito. Tuttavia, ci affidiamo solo a un gruppo di loro. Per una studio più completo basato su modelli gerarchici visitare il sito `https://www.leonardoegidi.com/covid-19`. Il set di dati contiene le seguenti variabili:

Nome variabile	Descrizione
`data`	Data di notifica
`stato`	Paese
`codice_regione`	Codice identificativo della regione (ISTAT 2019)
`denominazione_regione`	Nome della regione
`lat`	Latitudine
`long`	Longitudine
`ricoverati_con_sintomi`	Pazienti ospedalizzati con sintomi
`terapia_intensiva`	Pazienti in terapia intensiva
`totale_ospedalizzati`	Totale pazienti ospedalizzati
`isolamento_domiciliare`	Pazienti in isolamento domiciliare
`totale_positivi`	Totale dei pazienti attualmente positivi (ospedalizzati + domiciliari)
`variazione_totale_positivi`	Differenza giornaliera nel numero di attualmente positivi
`nuovi_positivi`	Nuovi casi giornalieri positivi
`dimessi_guariti`	Persone dimesse e guarite
`deceduti`	Morti da Covid-19
`totale_casi`	Totale dei casi positivi
`tamponi`	Tamponi effettuati
`casi_testati`	Numero totale di persone testate

Prima di tutto leggiamo i dati `.csv` con la funzione `read.csv2`:

```
covid <- read.csv2(file="covid19_ita.csv", sep=";")
```

ed estraiamo la variabile ICU:

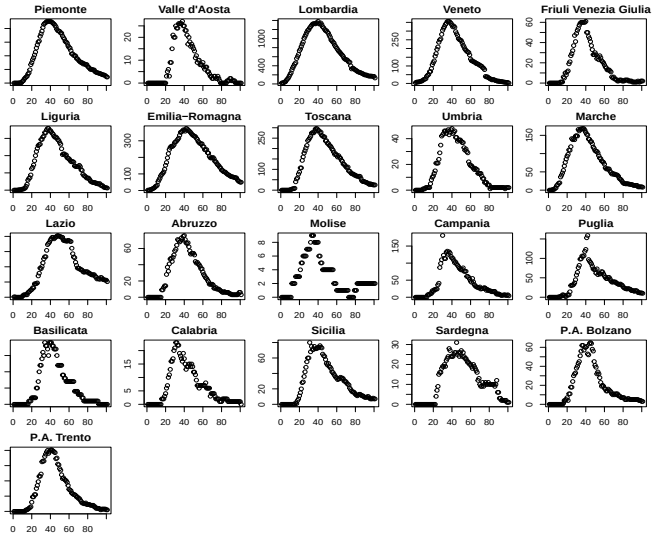
```
icu <- covid$terapia_intensiva
```

Per avere uno sguardo globale, possiamo rappresentare le serie temporali per ciascuna regione durante i 101 giorni:

```
reg <- c("Piemonte", "Valle d'Aosta", "Lombardia",
         "NA", "Veneto", "Friuli Venezia Giulia",
         "Liguria", "Emilia-Romagna", "Toscana",
         "Umbria", "Marche", "Lazio", "Abruzzo",
         "Molise", "Campania", "Puglia", "Basilicata",
         "Calabria", "Sicilia", "Sardegna",
         "P.A. Bolzano", "P.A. Trento")

par(mfrow= c(5,5), mar=c(2,1,2,2))
for (i in c(1:3, 5:22)){
```

```
plot(icu[covid$codice_regione==i],
     xlab = "Days", ylab = "ICU", main =as.character(reg)[i] )
}
```



Un modello di Poisson per le terapie intensive

Concentriamoci ora sulla terapia intensiva per una singola regione, ad esempio la Lombardia:

```
icu_lombardia <- icu[covid$denominazione_regione=="Lombardia"]
```

e proviamo a stimare una regressione di Poisson per questi dati, usando solo `days` come covariata:

```
days <- c(1:101)
mod_covid <- glm(icu_lombardia ~days,
                 family = "poisson")
summary(mod_covid)

##
## Call:
## glm(formula = icu_lombardia ~ days, family = "poisson")
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -39.26  -15.53   -1.49   13.93   23.43
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  6.7655443  0.0073371  922.10   <2e-16 ***
## days        -0.0059858  0.0001352  -44.28   <2e-16 ***
```

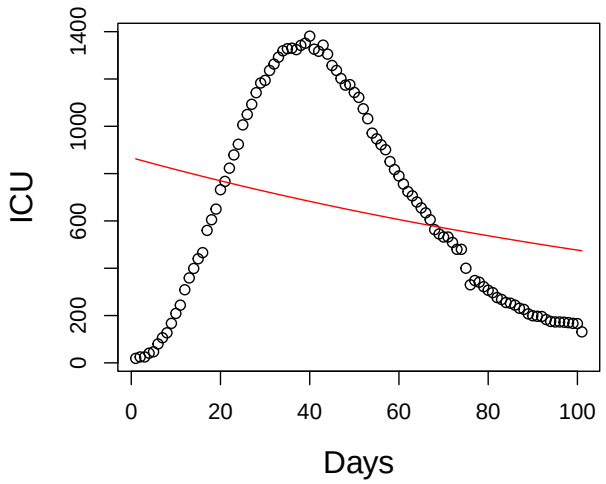
```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 32210  on 100  degrees of freedom
## Residual deviance: 30232  on  99  degrees of freedom
## AIC: 31041
##
## Number of Fisher Scoring iterations: 5
```

La covariata `days` è statisticamente diversa da zero. Per valutare la capacità predittiva di questo modello durante i giorni di osservazione, utilizziamo la funzione `predict`:

```
pred_lombardia <- predict(mod_covid, days = days,
                          type = "response")
```

Ora possiamo tracciare queste previsioni per ogni giorno insieme ai dati osservati:

```
par(mfrow=c(1,1))
par(mar=c(5,6,2,1))
plot(icu_lombardia, xlab = "Days", ylab ="ICU",
     cex.lab =1.4)
lines(pred_lombardia, col ="red")
```



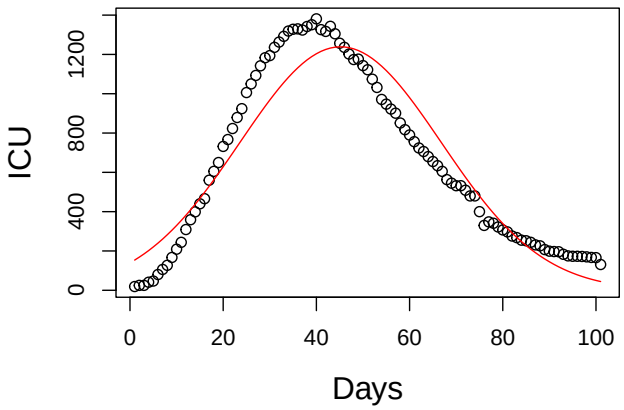
Il modello sembra non essere adatto a descrivere i dati: infatti le previsioni (linea rossa) non corrispondono ai dati a portata di mano...forse, l'aggiunta di alcuni predittori potrebbe aiutare.
La forma del grafico sopra suggerisce una dipendenza abbastanza non lineare tra `icu` e `days`: possiamo provare a includere un effetto quadratico nel glm:

```
mod_covid2 <- glm(icu_lombardia ~ days+I(days^2),
                  family = "poisson")
summary(mod_covid2)
```

```
##
## Call:
## glm(formula = icu_lombardia ~ days + I(days^2), family = "poisson")
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -14.8150   -5.2339   -0.9275    4.9976   12.8995
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.939e+00  1.716e-02   287.8   <2e-16 ***
## days         9.660e-02  7.590e-04   127.3   <2e-16 ***
## I(days^2)   -1.069e-03  7.789e-06  -137.2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 32210.4  on 100  degrees of freedom
## Residual deviance:  4115.8  on  98  degrees of freedom
## AIC: 4926.4
##
## Number of Fisher Scoring iterations: 5
```

Entrambi i coefficienti per `days` e il corrispondente effetto quadratico sono statisticamente diversi da zero; inoltre, la devianza residua del secondo modello è molto inferiore alla devianza del primo modello (4115.8 vs 30232!). Come prima, diamo un'occhiata alle previsioni:

```
pred_lombardia2<- predict(mod_covid2, days = days,
                          type = "response")
plot(icu_lombardia, xlab = "Days", ylab ="ICU",
     cex.lab =1.4)
lines(pred_lombardia2, col ="red")
```



Da questo grafico, confermiamo che questo secondo modello è decisamente migliore del primo.

Un modello di quasi verosimiglianza

Come notato sopra, un'inefficienza del modello di Poisson è che la media e la varianza coincidono: in tal modo, non è possibile catturare un'eventuale sovradispersione nei dati. Utilizzando la famiglia `quasipoisson` nella funzione `glm` possiamo stimare il parametro di dispersione tenendo conto della seguente specificazione:

$$\begin{aligned} \mathrm{E}(Y_i) =& \lambda_i \\ \mathrm{Var}(Y_i) =& \phi \mathrm{E}(Y_i) = \phi \lambda_i, \end{aligned}$$

dove ϕ è il cosiddetto parametro di dispersione che posso stimare con $\hat{\phi} = \frac{1}{n-p} \sum_i \frac{(y_i - \hat{\lambda}_i)^2}{\hat{\lambda}_i}$.

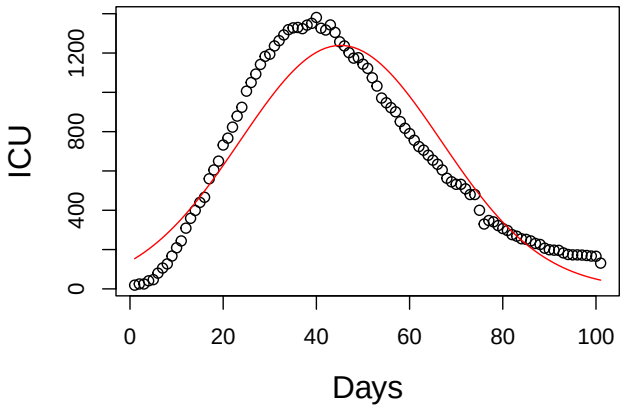
```
mod_covid_qp <- glm(icu_lombardia ~days+I(days^2),
                    family = "quasipoisson")
summary(mod_covid_qp)
```

```
##
## Call:
## glm(formula = icu_lombardia ~ days + I(days^2), family = "quasipoisson")
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -14.8150   -5.2339   -0.9275    4.9976   12.8995
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.939e+00  1.109e-01   44.53   <2e-16 ***
## days         9.660e-02  4.906e-03   19.69   <2e-16 ***
```

```
## I(days^2)   -1.069e-03  5.034e-05  -21.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 41.77469)
##
##     Null deviance: 32210.4  on 100  degrees of freedom
## Residual deviance:  4115.8  on  98  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

Come si può notare, le stime riportate nella prima colonna della tabella sono analoghe a quelle prodotte dal modello di Poisson, tuttavia cambiano gli standard errors associati a tali stime. La stima del parametro di dispersione è $\hat{\phi} = 41.8$: c'è quindi una chiara indicazione di sovradispersione. Rivediamo le previsioni:

```
pred_lombardia_qp<- predict(mod_covid_qp, days = days,
                            type = "response")
plot(icu_lombardia, xlab = "Days", ylab ="ICU",
     cex.lab =1.4)
lines(pred_lombardia_qp, col ="red")
```



Modello binomiale-negativa

Sempre per tenere debitamente in conto la sovradispersione, possiamo anche stimare direttamente un modello con binomiale negativa tramite la funzione `glm.nb` del pacchetto `MASS`:

```r
library(MASS)
mod_covid_nb <- glm.nb(icu_lombardia ~days+I(days^2))
summary(mod_covid_nb)
```

```
##
## Call:
## glm.nb(formula = icu_lombardia ~ days + I(days^2), init.theta = 6.398693746,
##     link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2260  -0.8113  -0.1877   0.7239   1.9113
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.580e+00  1.224e-01   37.42   <2e-16 ***
## days         1.024e-01  5.531e-03   18.52   <2e-16 ***
## I(days^2)   -1.040e-03  5.259e-05  -19.77   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(6.3987) family taken to be 1)
##
##     Null deviance: 438.14  on 100  degrees of freedom
## Residual deviance: 106.21  on  98  degrees of freedom
## AIC: 1355.5
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  6.399
##           Std. Err.:  0.920
##
##  2 x log-likelihood:  -1347.521
```

Come si può vedere, otteniamo valori per le stime simili a quelli precedentemente ottenuti tramite il Poisson e il quasi Poisson. Notiamo però come i valori di devianza residua e di AIC per questo ultimo modello siano estremamente più bassi di quelli ottenuti per Poisson e quasi Poisson (per quest'ultimo non si può però calcolare l'AIC, in quanto non si tratta di un approccio basato sulla verosimiglianza).

Adeguatezza dei modelli tramite i residui

Come strumento di confronto finale, possiamo tracciare i residui standardizzati per i quattro modelli trattati: Poisson con solo il tempo; Poisson, quasi Poisson e binomiale-negativa con anche l'effetto

quadratico del tempo. Per i modelli di Poisson i residui standardizzati sono:

$$r_i = \frac{y_i - \hat{\lambda}_i}{\sqrt{\hat{\lambda}_i}},\ i = 1, \ldots, N,$$

dove $\hat{\lambda}_i = \exp\{\hat{\beta}_0 + \hat{\beta}_1 days_i + \hat{\beta}_2 days_i^2\}$. Per il modello quasi Poisson i residui standardizzati sono:

$$r_i = \frac{y_i - \hat{\lambda}_i}{\sqrt{\hat{\phi}\hat{\lambda}_i}},\ i = 1, \ldots, N,$$

dove $\hat{\phi}$ è la stima del parametro di dispersione.
Per quanto riguarda invece i residui della binomiale negativa abbiamo:

$$r_i = \frac{y_i - \hat{\lambda}_i}{\sqrt{\hat{\lambda}_i + \hat{\lambda}_i^2/\hat{\phi}}},\ i = 1, \ldots, N,$$

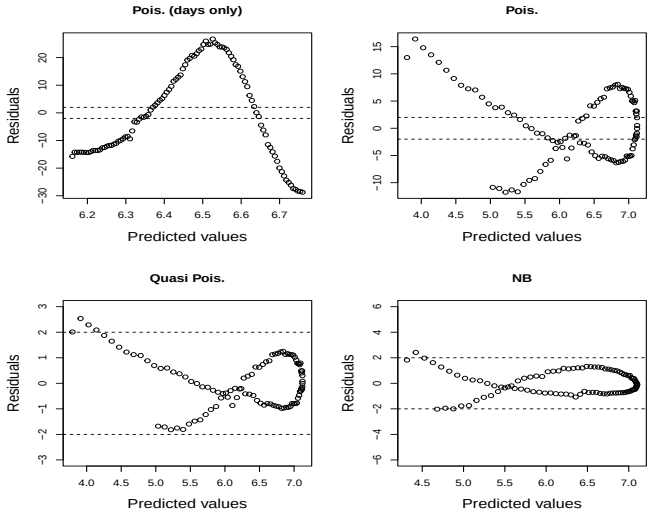
dove $\hat{\phi}$ è la stima del parametro di dispersione della binomiale negativa: notare che tale parametro ha un diverso significato rispetto a quanto avviene nel quasi Poisson. Nell'esempio sopra per il modello BN troviamo $\hat{\phi} = 6.4$, (la varianza della BN è $\lambda + \lambda^2/\phi$), mentre nel quasi Poisson troviamo $\hat{\phi} = 41.8$ (ma qui la varianza è $\phi\lambda$). Estraiamo i residui dai quattro modelli:

```
# estraggo residui dai modelli e calcolo le versioni standardizzate
res1 <- (icu_lombardia-mod_covid$fitted.values)/sqrt(mod_covid$fitted.values)
res2 <- (icu_lombardia-mod_covid2$fitted.values)/sqrt(mod_covid2$fitted.values)
res3 <- (icu_lombardia-mod_covid_qp$fitted.values)/sqrt(41.8*mod_covid_qp$fitted.values)
res4 <- (icu_lombardia-mod_covid_nb$fitted.values)/sqrt(mod_covid_nb$fitted.values +
                                    (1/6.4)* (mod_covid_nb$fitted.values)^2)
```

E ora rappresentiamoli

```
par(mfrow=c(2,2))
plot(log(mod_covid$fitted.values), res1,
     xlab = "Predicted values",
     ylab = "Residuals", cex.lab =1.4, main ="Pois. (days only)")
abline(h =-2, lty=2)
abline(h=2, lty=2)
plot(log(mod_covid2$fitted.values), res2,
     xlab = "Predicted values",
     ylab = "Residuals", cex.lab =1.4, main = "Pois.")
abline(h =-2, lty=2)
abline(h=2, lty=2)
plot(log(mod_covid_qp$fitted.values), res3,
     xlab = "Predicted values",
     ylab = "Residuals", cex.lab =1.4, ylim=c(-3,3), main = "Quasi Pois.")
abline(h =-2, lty=2)
```

```
abline(h=2, lty=2)
plot(log(mod_covid_nb$fitted.values), res4,
     xlab = "Predicted values",
     ylab = "Residuals", cex.lab =1.4, ylim=c(-6,6), main = "NB")
abline(h =-2, lty=2)
abline(h=2, lty=2)
```



Molti dei residui standardizzati nel modello quasi Poisson e in quello binomiale negativo sono limitati tra -2 e 2: questo suggerisce e conferma chiaramente un adattamento migliore di questi modelli rispetto al semplice modello di Poisson.