Negative Binomial Regression

Proudly copied from: https://georgederpa.github.io/teaching/countModels.html

$$y_i \sim NB(p_i, r) p_i = \frac{r}{r+\lambda_i} \log \lambda_i = \mu_i \mu_i = \beta \mathbf{X}_i \beta_j \sim \mathcal{N}(0, 0.0001) r \sim \mathcal{U}(0, 50)$$

Load Data

```
rm(list=ls())
setwd("~/Downloads/BLMS/BLMS")
library(rjags)
```

```
## Loading required package: coda
```

```
## Linked to JAGS 4.3.1
```

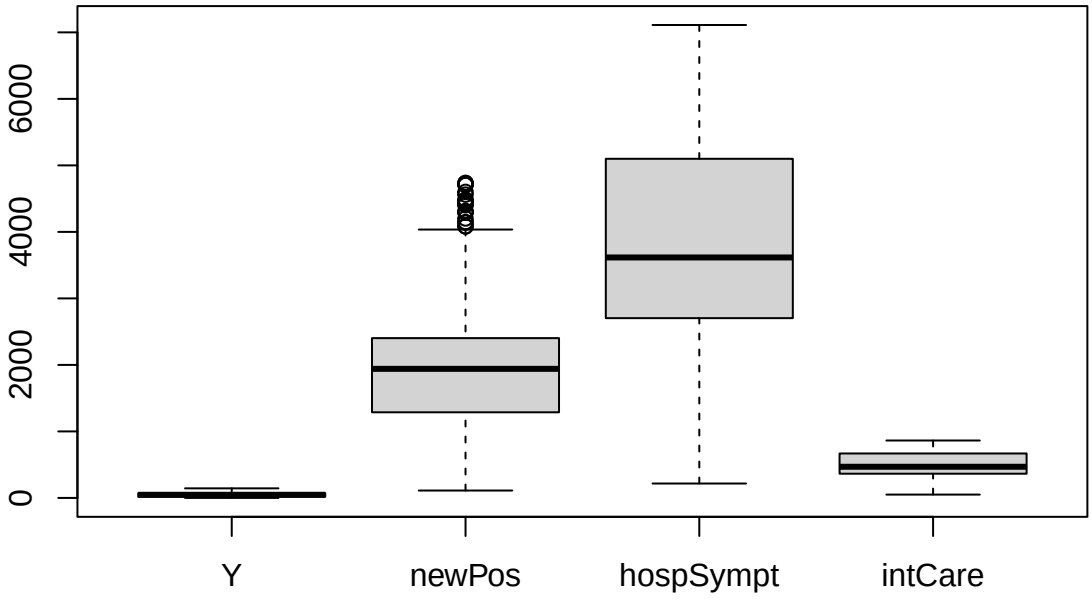
```
## Loaded modules: basemod,bugs
```

```
dataset = readRDS("dataset.rds")

newPos = dataset$newpos_av7D
hospSympt = dataset$hospitalized_with_symptoms_av7D
intCare = dataset$intensive_care_av7D
deathsH8 = dataset$deathsH8

Y = deathsH8
X = cbind(newPos, hospSympt, intCare)
data = cbind(Y, X)

boxplot(data)
```



JAGS Model

```
model_string=textConnection("model{
    ## Likelihood
```

```
    for(i in 1:N){
      Y[i] ~ dnegbin(p[i],r)
      p[i] <- r/(r+lambda[i])
      log(lambda[i]) <- mu[i]
      mu[i] <- inprod(beta[], X[i,])
    }
    ## Priors
    for(j in 1:ncol){
      beta[j] ~ dnorm(0, 0.0001)
    }
    r ~ dunif(0,50)
}")

N = length(Y)
ncol = ncol(X)
dataList = list(Y=Y, X=X, N=N, ncol=ncol)

jagsModel = jags.model(model_string, data=dataList, n.chains = 1)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 212
##    Unobserved stochastic nodes: 4
##    Total graph size: 1918
##
## Initializing model
```

```
update(jagsModel, 5000)
out = coda.samples(jagsModel, variable.names = c("beta","r"), n.iter = 10000, thin=5)

summary(out)
```

```
##
## Iterations = 6005:16000
## Thinning interval = 5
## Number of chains = 1
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean        SD  Naive SE Time-series SE
## beta[1] 0.0002398 0.0001188 2.657e-06      1.016e-05
## beta[2] 0.0004552 0.0001405 3.141e-06      1.999e-05
## beta[3] 0.0026835 0.0008047 1.799e-05      8.623e-05
## r       1.4259321 0.1354845 3.030e-03      2.924e-03
##
## 2. Quantiles for each variable:
##
##              2.5%       25%       50%       75%     97.5%
## beta[1] 1.149e-05 0.0001591 0.0002386 0.0003195 0.0004631
## beta[2] 1.622e-04 0.0003581 0.0004669 0.0005584 0.0007081
```

```
## beta[3] 1.286e-03 0.0020934 0.0026278 0.0032335 0.0043293
## r       1.179e+00 1.3319789 1.4206480 1.5152393 1.7086603
```

```
par(mar=c(1,1,1,1))
plot(out)
```