Lab 12: Titanic data

Wayne Stewart

2018-04-11 10:15:54

The course

As you have noticed "the Titanic" has been a theme of the course. The course has been structured into three parts:

* Distributional results and basics of R
* Binomial: a simple model where Bayesian methodology is learnt
* The GLM: A more advanced application of Bayesian theory

All the skills that you have learnt in parts 1 and 2 will now be applied to a logistic regression which is a special case of the GLM.

We will now start to analyze the Titanic data set and you will perfect this in Assignment 4.

Task 0: The story

Summarize the Titanic story by reading the following web page: (https://www.history.com/this-day-in-history/unsinkable-titanic-sinks)

You only need a paragraph!!

Task 1: The Titanic data set

We will use the data set as prepared in the `vcdExtra` package. We will aslo use the `gpairs` package. Please install.

Make a pairs plot of the data using `gpairs()`

```
library(vcdExtra)
library(gpairs)
data("Titanicp")
head(Titanicp)
```

```
##   pclass survived    sex     age sibsp parch
## 1    1st survived female 29.0000     0     0
## 2    1st survived   male  0.9167     1     2
## 3    1st     died female  2.0000     1     2
## 4    1st     died   male 30.0000     1     2
## 5    1st     died female 25.0000     1     2
## 6    1st survived   male 48.0000     0     0
```

Notice that there are a number of categorical variables and a continuous variable `age`.

Using the R help for the package describe the variables in the Titanicp data set.

Task 2: Interpreting the plots

Interpret the plots below:

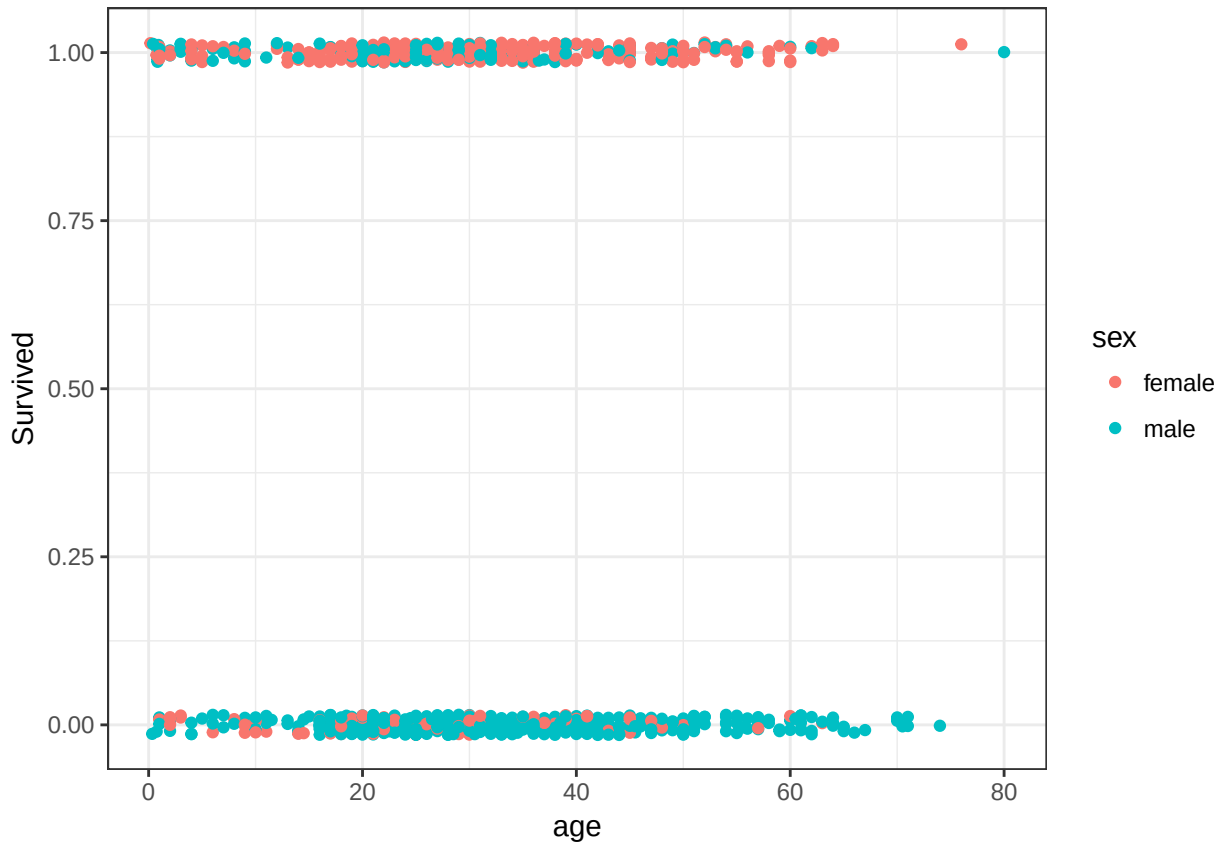
```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```
g = ggplot(Titanicp, aes(x = age, y=as.numeric(survived =="survived"),color = sex)) + ylab("Survived") +

g = g + geom_point(position = position_jitter(height = 0.015, width =0))
g
```

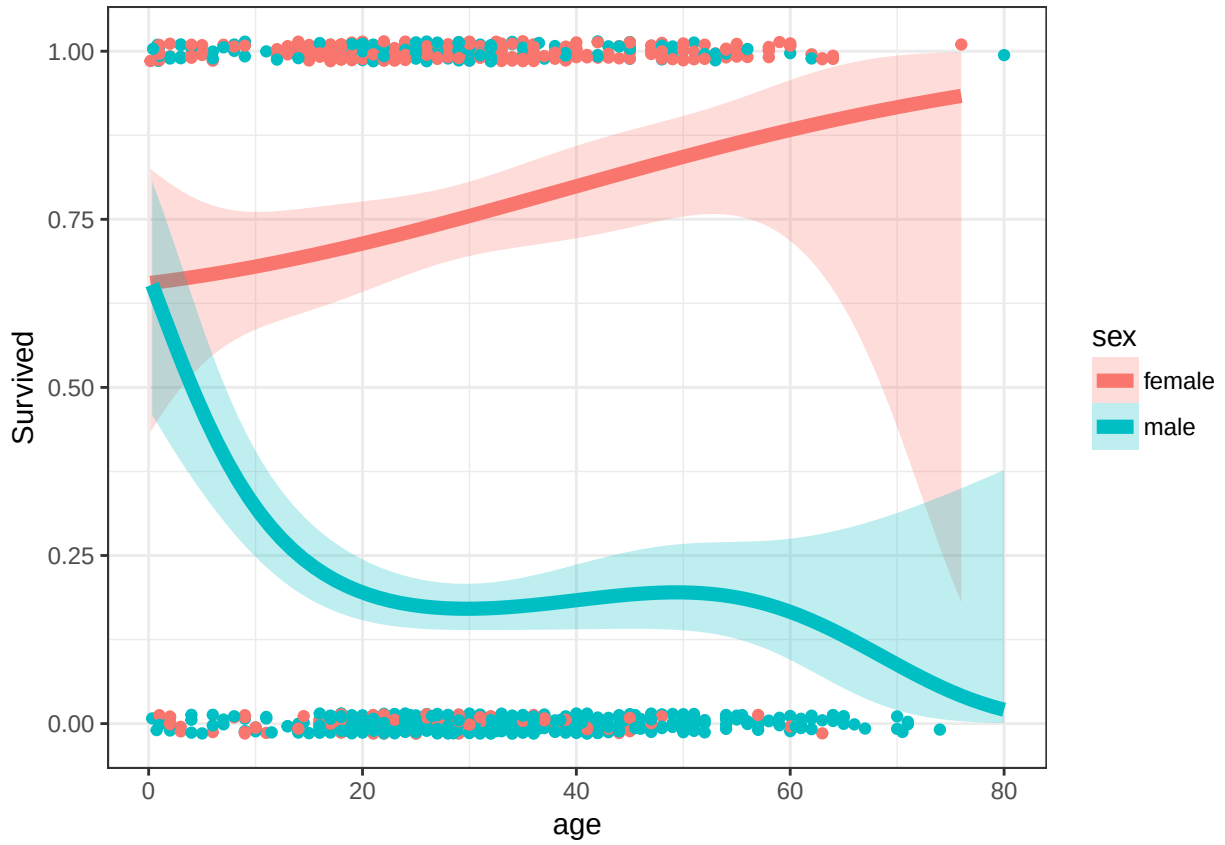
```
## Warning: Removed 263 rows containing missing values (geom_point).
```




```
g = g + stat_smooth(method = "glm", method.args = list(family=binomial("logit")), formula = y ~ x + I(x
g
```

```
## Warning: Removed 263 rows containing non-finite values (stat_smooth).
```

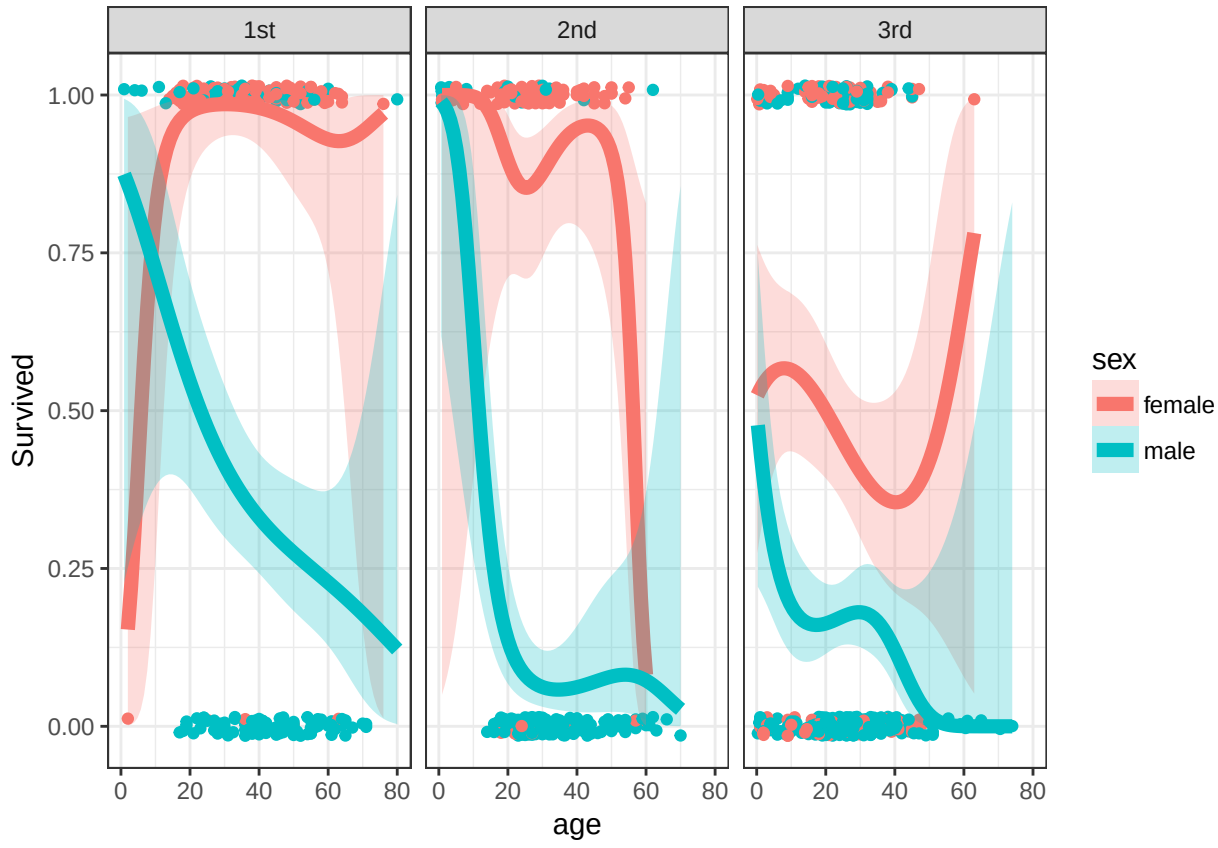
```
## Warning: Removed 263 rows containing missing values (geom_point).
```




```
g = g + facet_wrap(~pclass)
g
```

```
## Warning: Removed 263 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 263 rows containing missing values (geom_point).
```



Task 3: Classical analysis using `glm()`

We will perform a logistic regression using glm.

```
clglm = glm(survived ~ sex + age + sex:age,family = "binomial", data = Titanicp)
summary(clglm)
```

```
##
## Call:
## glm(formula = survived ~ sex + age + sex:age, family = "binomial",
##     data = Titanicp)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0247  -0.7158  -0.5776   0.7707   2.2960
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.493381   0.254188   1.941 0.052257 .
## sexmale     -1.154139   0.339337  -3.401 0.000671 ***
## age          0.022516   0.008535   2.638 0.008342 **
## sexmale:age -0.046276   0.011216  -4.126 3.69e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1414.6  on 1045  degrees of freedom
## Residual deviance: 1083.4  on 1042  degrees of freedom
##   (263 observations deleted due to missingness)
## AIC: 1091.4
##
## Number of Fisher Scoring iterations: 4
```

What are the classical point estimates?

Task 4: Use the classical model to make data for Jags

Complete the code below (one line, y=)

Why not just use the original data?

```
mat1=model.matrix(clglm)
mat2=model.frame(clglm)
head(mat1)
head(mat2)

y = with(mat2, ifelse(survived == "survived", ..., ...))

dataList=list(y = y, x = mat1[, "age"],sexm = mat1[,"sexmale"], sexmx = mat1[,"sexmale:age"] , n = leng
```

Task 5: Now we will use the classical estimates as initial values in a jags script

Complete the Jags script below

Warning

The MCMC sampler is very sensitive to initial values. Within each gibbs iteration Jags will choose a slice sampler – this will take some time. You may need to wait a few minutes for the sampling to complete.

```
library(rjags)

#Define the model:
modelString = "
model{
  for(i in 1:n){
 ...

  }
for(j in 1:4){
  beta[j] ~ dnorm(0,1.0E-3)
}
}
```

```
"
writeLines( modelString , con="TEMPmodel.txt" )

# close quote for modelStri
#  initsList = list( theta=thetaInit )

initsList = list(beta = c(0.5,0.02,-1.15,-0.05))
# Run the chains:
jagsModel = jags.model( file="TEMPmodel.txt" , data=dataList , inits=initsList ,
                        n.chains=3 , n.adapt=500 )
list.samplers(jagsModel)

update( jagsModel , n.iter=500 )
codaSamples = coda.samples( jagsModel , variable.names=c("beta"),
                            n.iter=33340 )
save( codaSamples , file=paste0("lab12","Mcmc.Rdata") )



library(ggmcmc)
s = ggs(codaSamples)
d=ggs_density(s)

print(d)

cr =  ggs_crosscorrelation(s)
print(cr)

summary(codaSamples)
```

Task 6: Interpretation

Interpret all the Bayesian output

* Interpret the point estimates for the betas
* Interpret the interval estimates for the betas
* How do you know the MCMC sampler converged to stationarity?
* Compare your results with the classical analysis estimates