

# DSA 8020 R Session 7: Logistic Regression and Poisson Regression

Whitney

## Contents

Logistic Regression: Horseshoe Crab Malting . . . . .	1
Load the data . . . . .	1
Fit a Linear Regression . . . . .	2
Fit a Logistic Regression . . . . .	2
Confidence Intervals . . . . .	3
Prediction . . . . .	4
Raw Residual plot . . . . .	5
Binned Residuals . . . . .	6
Model selection . . . . .	6
Generalized additive logistic regression . . . . .	7
Poisson Regression . . . . .	8
Flying-Bomb Hits on London During World War II [Clarke, 1946; Feller, 1950] . . . . .	8
US Landfalling Hurriances . . . . .	9
Load the environmetal variables . . . . .	10
Plot hurricane counts against environmental variables . . . . .	10
Linear Regression . . . . .	11
Poisson Regression . . . . .	11
Generalized additive Poisson regression . . . . .	12

## Logistic Regression: Horseshoe Crab Malting

*Data Source:* Brockmann, H. J. (1996). Satellite male groups in horseshoe crabs, *Limulus polyphemus*. *Ethology*, 102(1), 1-21.

### Load the data

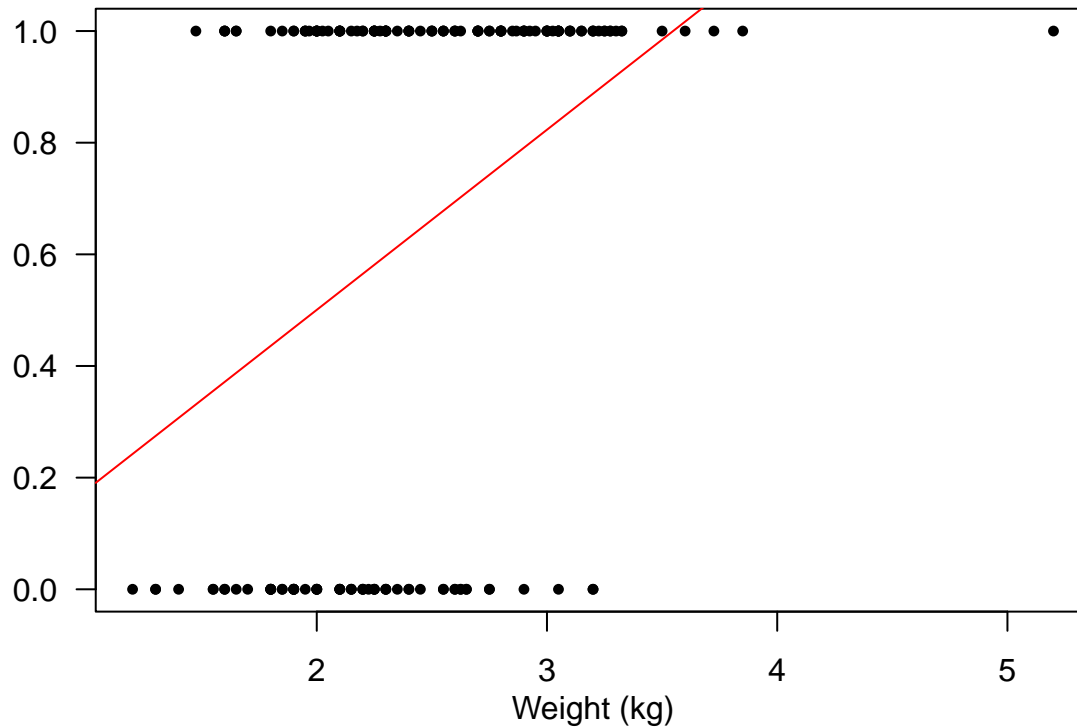
This dataset is obtained from the website of *Alan Agresti*, Distinguished Professor Emeritus at the University of Florida.

```
crab <- read.table("http://users.stat.ufl.edu/~aa/cda/data/Crabs.dat", header = T)
```

## Fit a Linear Regression

Let's fit a simple linear regression using weight as the predictor

```
lmFit <- lm(y ~ weight, data = crab)
par(mar = c(3.5, 3.5, 0.8, 0.6))
with(crab, plot(weight, y, pch = 16, cex = 0.75, las = 1, xlab = "", ylab = ""))
mtext("Weight (kg)", side = 1, line = 2)
abline(lmFit, col = "red")
```



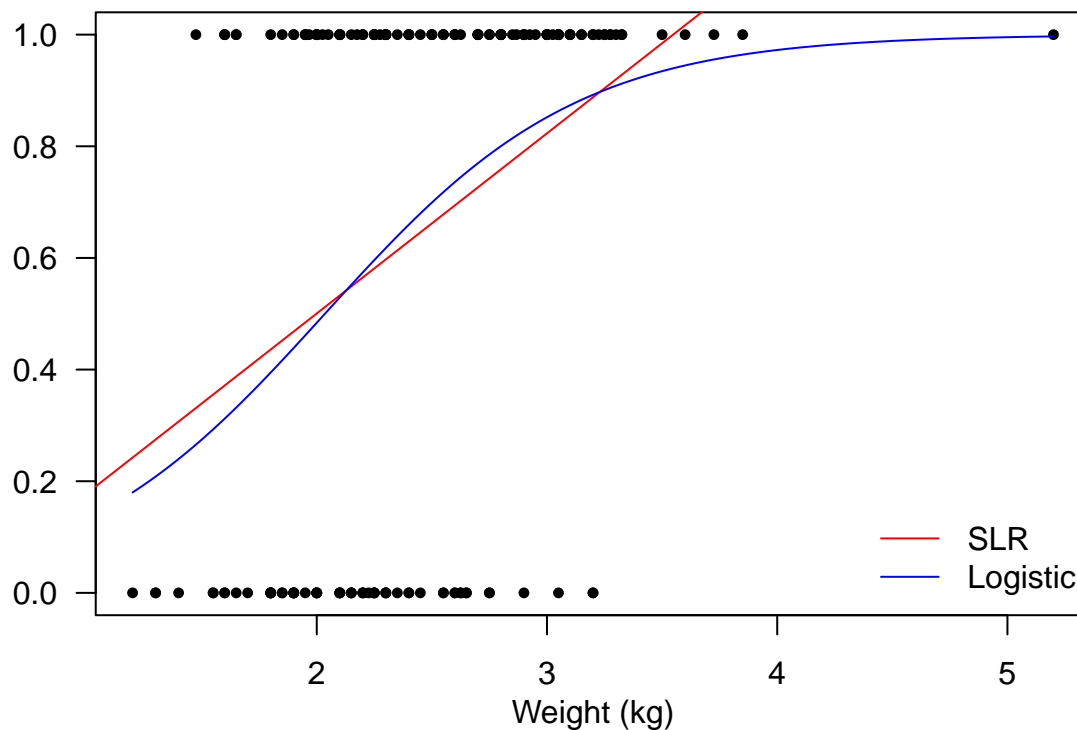
## Fit a Logistic Regression

```
logitFit <- glm(y ~ weight, data = crab, family = "binomial")
summary(logitFit)
```

```
##
## Call:
## glm(formula = y ~ weight, family = "binomial", data = crab)
##
## Deviance Residuals:
##   Min       1Q   Median       3Q      Max
## -2.1108  -1.0749   0.5426   0.9122   1.6285
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.6947     0.8802  -4.198 2.70e-05 ***
## weight         1.8151     0.3767   4.819 1.45e-06 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 195.74  on 171  degrees of freedom
## AIC: 199.74
##
## Number of Fisher Scoring iterations: 4
```

```
# Plot the fits
rg <- range(crab$weight)
xg <- seq(rg[1], rg[2], 0.01)
pred <- predict(logitFit, newdata = data.frame(weight = xg), type = "response")
par(mar = c(3.5, 3.5, 0.8, 0.6))
plot(crab$weight, crab$y, pch = 16, cex = 0.75, las = 1, xlab = "", ylab = "")
mtext("Weight (kg)", side = 1, line = 2)
abline(lmFit, col = "red"); lines(xg, pred, col = "blue")
legend("bottomright", legend = c("SLR", "Logistic"),
      col = c("red", "blue"), lty = 1, bty = "n")
```



### Confidence Intervals

```
# Normal approximation
est <- summary(logitFit)$coefficients
est
```

```
## Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept) -3.694726  0.8801975 -4.197611  2.697457e-05
## weight      1.815145  0.3766959  4.818594  1.445736e-06
```

```
(CI_norm <- est[2, 1] + c(-1, 1) * qnorm(0.975) * est[2, 2])
```

```
## [1] 1.076834 2.553455
```

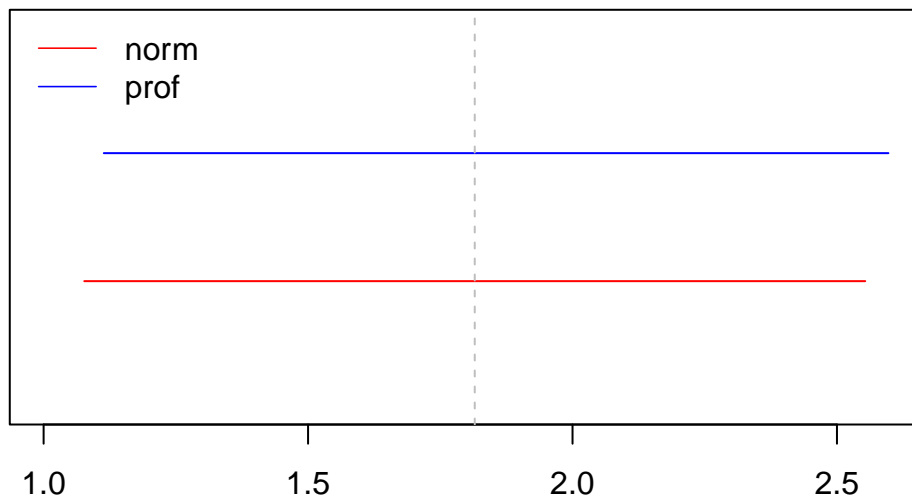
```
# Profile likelihood CI
library(MASS)
(CI_prof <- confint(logitFit)[2,])
```

```
## Waiting for profiling to be done...
```

```
## 2.5 % 97.5 %
## 1.113790 2.597305
```

```
plot(1, type = "n", xlab = "", ylab = "", xlim = c(1, 2.6), ylim = c(-0.05, 0.1),
     yaxt = "n", main = expression(hat(beta[1])))
segments(CI_norm[1], 0, CI_norm[2], col = "red")
segments(CI_prof[1], 0.05, CI_prof[2], col = "blue")
abline(v = est[2, 1], lty = 2, col = "gray")
legend("topleft", legend = c("norm", "prof"),
      col = c("red", "blue"), lty = 1, lwd = 0.8, bty = "n")
```

$$\hat{\beta}_1$$



## Prediction

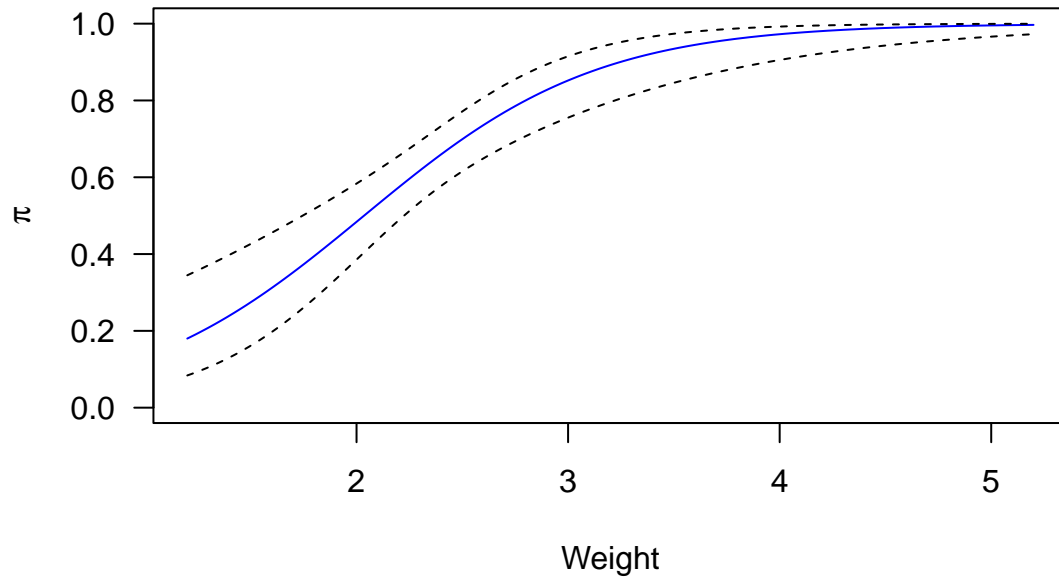
```
pred <- predict(logitFit, newdata = data.frame(weight = xg), se.fit = TRUE)
fit <- pred$fit; se <- pred$se.fit

plot(xg, exp(fit) / (1 + exp(fit)), type = "l", col = "blue", ylim = c(0, 1),
```

```

las = 1, xlab = "Weight", ylab = expression(pi))
lines(xg, exp(fit + 1.96 * se) / (1 + exp(fit + 1.96 * se)), lty = 2)
lines(xg, exp(fit - 1.96 * se) / (1 + exp(fit - 1.96 * se)), lty = 2)

```



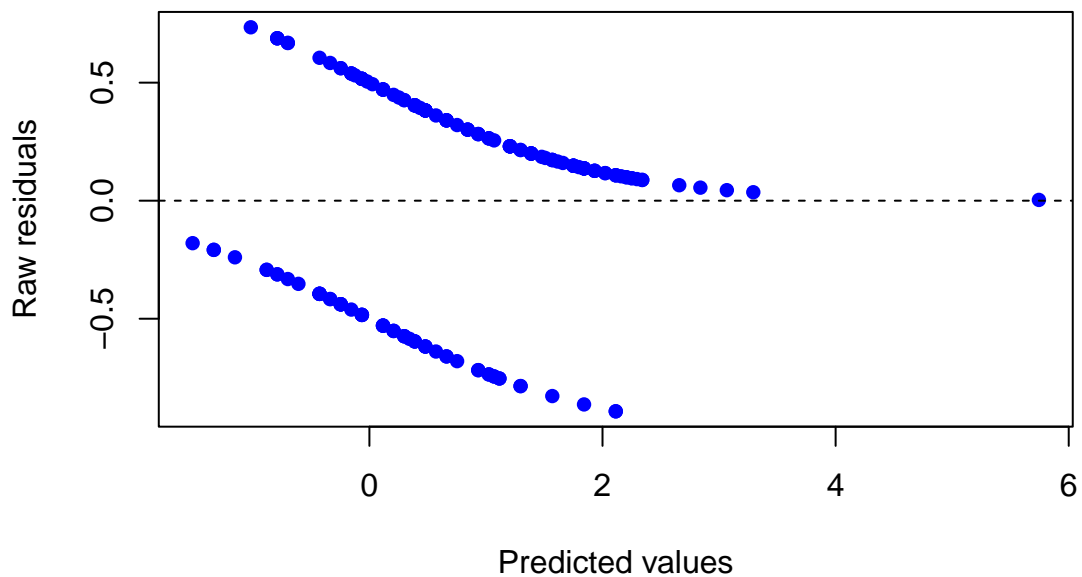
### Raw Residual plot

```

res <- resid(logitFit, type = "response")
pred <- predict(logitFit)

plot(pred, res, col = "blue", pch = 16, xlab = "Predicted values", ylab = "Raw residuals")
abline(h = 0, lty = 2)

```

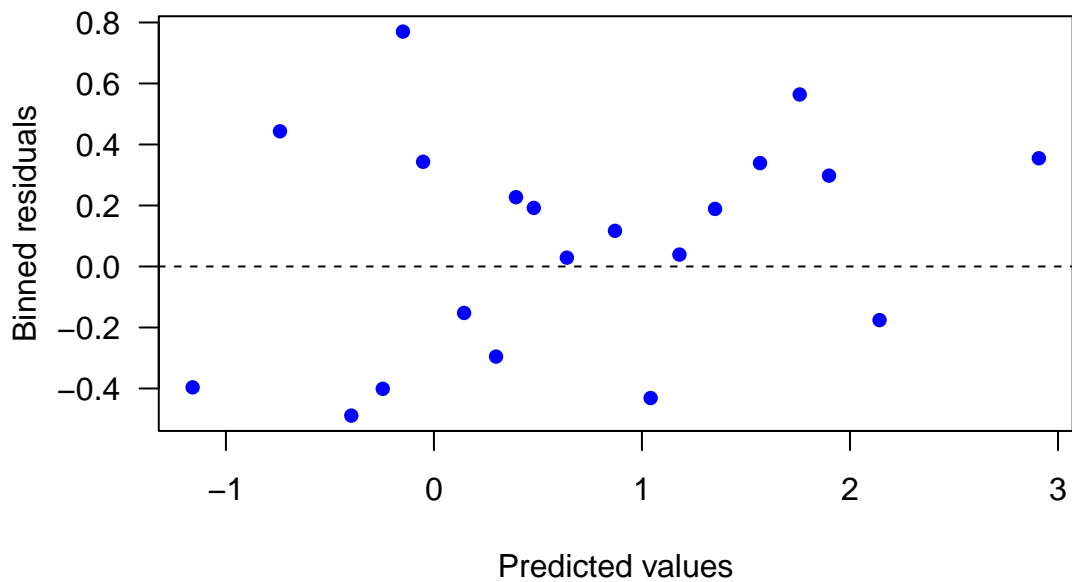


## Binned Residuals

```
breaks <- quantile(crab$weight, seq(0, 1, length.out = 20 + 1))
wt_bin <- findInterval(crab$weight, breaks, rightmost.closed = TRUE)
library(dplyr)
crab.res <- mutate(crab, res = residuals(logitFit), Lpred = predict(logitFit), bin = wt_bin)

res_bin <- tapply(crab.res$res, crab.res$bin, mean)
Lpred_bin <- tapply(crab.res$Lpred, crab.res$bin, mean)

plot(res_bin ~ Lpred_bin, xlab = "Predicted values",
      ylab = "Binned residuals", col = "blue", pch = 16, las = 1)
abline(h = 0, lty = 2)
```



## Model selection

```
logitFit2 <- glm(y ~ weight + width, data = crab, family = "binomial")
summary(logitFit2)
```

```
##
## Call:
## glm(formula = y ~ weight + width, family = "binomial", data = crab)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.1127  -1.0344   0.5304   0.9006   1.7207
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -9.3547     3.5280  -2.652  0.00801 **
## weight         0.8338     0.6716   1.241  0.21445
```

```
## width          0.3068      0.1819   1.686  0.09177 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 225.76 on 172 degrees of freedom
## Residual deviance: 192.89 on 170 degrees of freedom
## AIC: 198.89
##
## Number of Fisher Scoring iterations: 4
```

```
step(logitFit2)
```

```
## Start: AIC=198.89
## y ~ weight + width
##
##           Df Deviance   AIC
## - weight  1   194.45 198.45
## <none>      192.89 198.89
## - width   1   195.74 199.74
##
## Step: AIC=198.45
## y ~ width
##
##           Df Deviance   AIC
## <none>      194.45 198.45
## - width   1   225.76 227.76

##
## Call: glm(formula = y ~ width, family = "binomial", data = crab)
##
## Coefficients:
## (Intercept)      width
##   -12.3508      0.4972
##
## Degrees of Freedom: 172 Total (i.e. Null); 171 Residual
## Null Deviance:      225.8
## Residual Deviance: 194.5   AIC: 198.5
```

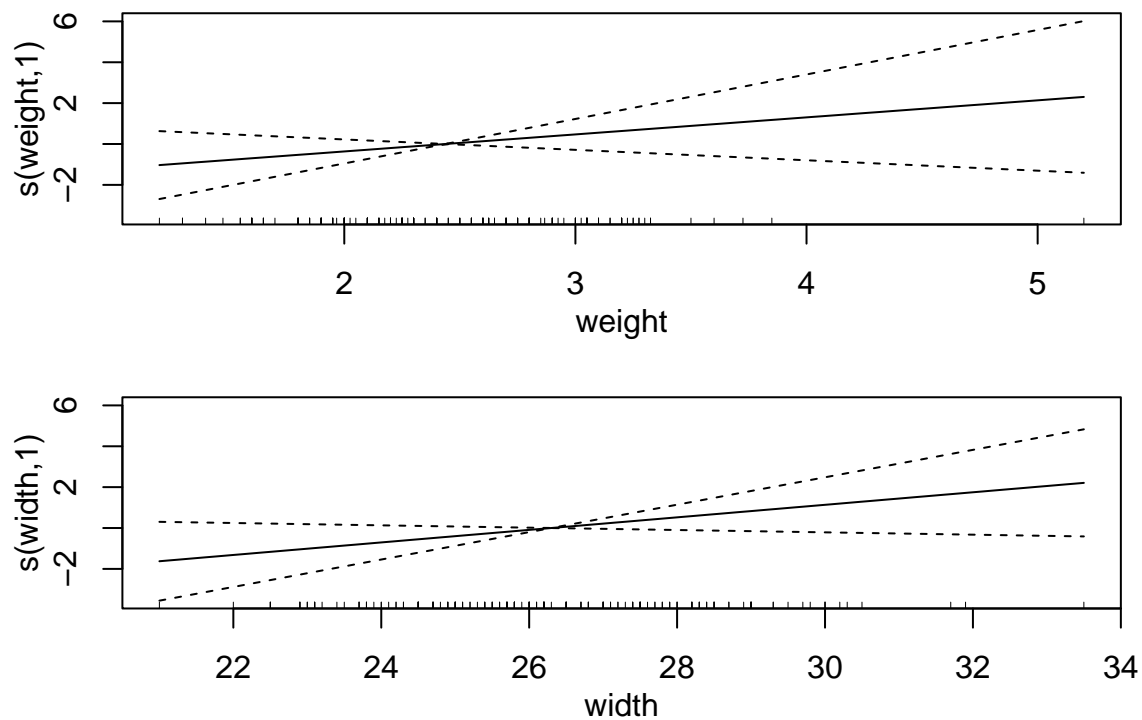
### Generalized additive logistic regression

```
library(mgcv)
logit_gam <- gam(y ~ s(weight) + s(width), family = "binomial", data = crab)
summary(logit_gam)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
```

```
## y ~ s(weight) + s(width)
##
## Parametric coefficients:
##           Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.7456    0.1847   4.036 5.43e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##           edf Ref.df Chi.sq p-value
## s(weight)   1     1  1.541  0.2145
## s(width)    1     1  2.843  0.0918 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.162   Deviance explained = 14.6%
## UBRE = 0.14966   Scale est. = 1           n = 173
```

```
par(mfrow = c(2, 1), mar = c(3.5, 3.5, 1, 0.5), mgp = c(2, 1, 0))
plot(logit_gam)
```



## Poisson Regression

Flying-Bomb Hits on London During World War II [Clarke, 1946; Feller, 1950]

```
count <- c(229, 211, 93, 35, 7, 1)
grids <- 576
hits <- 537
```



```
lambda <- hits / grids
count_expected <- c(grids * dpois(0:4, lambda = lambda),
                    grids * ppois(4, lambda = lambda, lower.tail = F))
round(count_expected, 1)
```

```
## [1] 226.7 211.4 98.5 30.6 7.1 1.6
```

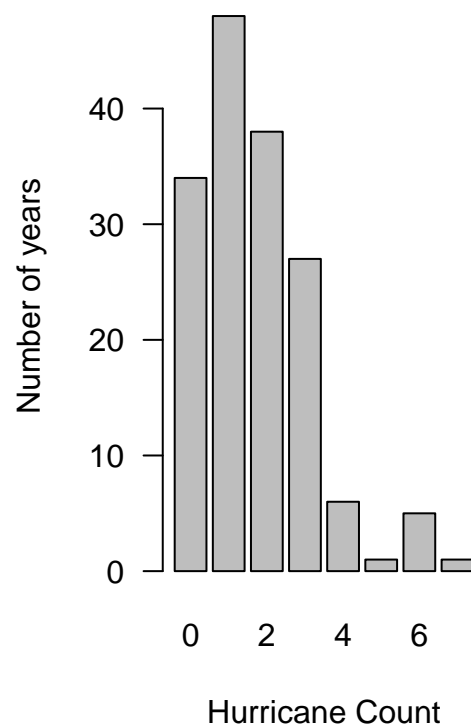
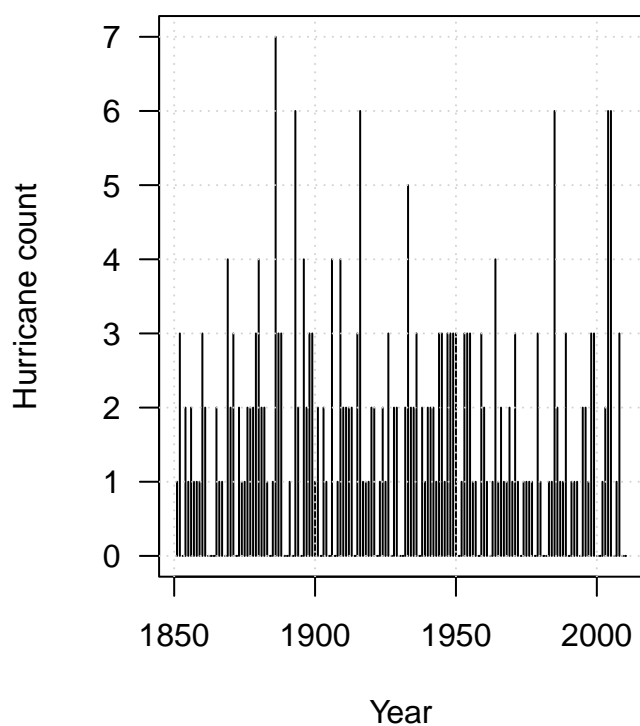
## US Landfalling Hurriances

This dataset is courtesy of *James Elsner*, Earl B. and Sophia H. Shaw Professor in the Department of Geography at Florida State University.

```
# load the hurriance count
con = "http://myweb.fsu.edu/jelsner/Book/Chap07/US.txt"
hurricanes = read.table(con, header = T)
head(hurricanes)
```

```
##   Year All MUS G FL E
## 1 1851  1  1 0  1 0
## 2 1852  3  1 1  2 0
## 3 1853  0  0 0  0 0
## 4 1854  2  1 1  0 1
## 5 1855  1  1 1  0 0
## 6 1856  2  1 1  1 0
```

```
par(las = 1, mar = c(4.6, 3.9, 0.8, 0.6))
layout(matrix(c(1, 2), 1, 2, byrow = TRUE), widths = c(0.57, 0.43))
plot(hurricanes$Year, hurricanes$All, type = "h", xlab = "Year", ylab = "Hurricane count")
grid()
barplot(table(hurricanes$All), xlab = "Hurricane Count", ylab = "Number of years", main = "")
```



## Load the environmental variables

```
load("annual.RData")
data <- data.frame(All = hurricanes$All, SOI = annual$soi, NAO = annual$nao, SST = annual$sst,
                  SSN = annual$ssn)
data <- data[-(1:15),]
```

## Plot hurricane counts against environmental variables

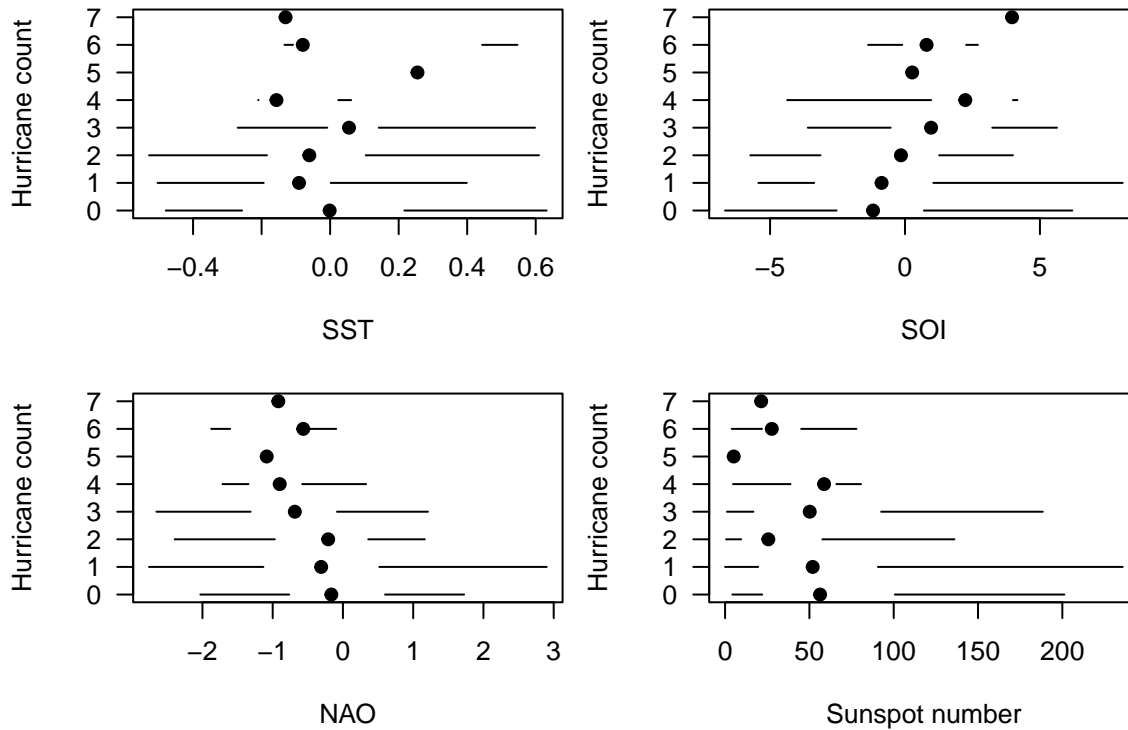
```
H <- hurricanes

par(mfrow = c(2, 2), mar = c(4.5, 4, 1, 0.6))
plot(range(annual$sst, na.rm = TRUE), c(0, 7), type = "n", ylab = "Hurricane count", xlab = "SST",
      las = 1)
for(i in 0:7){
  points(fivenum(annual$sst[H$All == i])[3], i, pch = 19)
  lines(c(fivenum(annual$sst[H$All == i])[1], fivenum(annual$sst[H$All == i])[2]), c(i, i))
  lines(c(fivenum(annual$sst[H$All == i])[4], fivenum(annual$sst[H$All == i])[5]), c(i, i))
}
plot(range(annual$soi, na.rm = TRUE), c(0, 7), type = "n", ylab = "Hurricane count", xlab = "SOI",
      las = 1)

for(i in 0:7){
  points(fivenum(annual$soi[H$All == i])[3], i, pch=19)
  lines(c(fivenum(annual$soi[H$All == i])[1], fivenum(annual$soi[H$All == i])[2]), c(i, i))
  lines(c(fivenum(annual$soi[H$All == i])[4], fivenum(annual$soi[H$All == i])[5]), c(i, i))
}
plot(range(annual$nao, na.rm = TRUE), c(0, 7), type = "n", ylab = "Hurricane count", xlab = "NAO",
      las = 1)

for(i in 0:7){
  points(fivenum(annual$nao[H$All == i])[3], i, pch=19)
  lines(c(fivenum(annual$nao[H$All == i])[1], fivenum(annual$nao[H$All == i])[2]), c(i, i))
  lines(c(fivenum(annual$nao[H$All == i])[4], fivenum(annual$nao[H$All == i])[5]), c(i, i))
}
plot(range(annual$ssn, na.rm = TRUE), c(0, 7), type = "n", ylab = "Hurricane count",
      xlab = "Sunspot number", las = 1)

for(i in 0:7){
  points(fivenum(annual$ssn[H$All == i])[3], i, pch = 19)
  lines(c(fivenum(annual$ssn[H$All == i])[1], fivenum(annual$ssn[H$All == i])[2]), c(i, i))
  lines(c(fivenum(annual$ssn[H$All == i])[4], fivenum(annual$ssn[H$All == i])[5]), c(i, i))
}
```



### Linear Regression

```
lmFull <- lm(All ~ ., data = data)
predict(lmFull, newdata = data.frame(SOI = -3, NAO = 3, SST = 0, SSN = 250))
```

```
##          1
## -0.318065
```

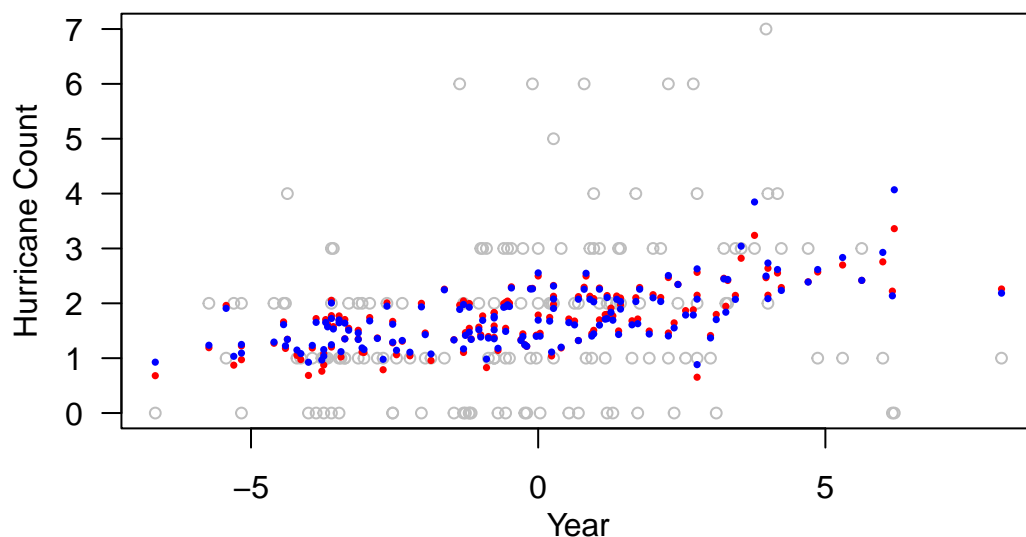
### Poisson Regression

```
PoiFull <- glm(All ~ ., data = data, family = "poisson")
summary(PoiFull)
```

```
##
## Call:
## glm(formula = All ~ ., family = "poisson", data = data)
##
## Deviance Residuals:
##   Min       1Q   Median       3Q      Max
## -2.8530  -0.8984  -0.1376   0.6027   2.4720
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.595288   0.103342   5.760 8.39e-09 ***
## SOI          0.061863   0.021319   2.902 0.00371 **
## NAO         -0.166595   0.064427  -2.586 0.00972 **
```

```
## SST          0.228972  0.255289  0.897  0.36977
## SSN          -0.002306  0.001372 -1.681  0.09284 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
## Null deviance: 197.89  on 144  degrees of freedom
## Residual deviance: 174.81  on 140  degrees of freedom
## AIC: 479.64
##
## Number of Fisher Scoring iterations: 5
```

```
plot(data$SOI, hurricanes$All[-(1:15)], cex = 0.75, col = "gray",
      xlab = "", ylab = "", las = 1)
mtext("Hurricane Count", side = 2, line = 2)
mtext("Year", side = 1, line = 2)
points(data$SOI, predict(lmFull), col = "red", cex = 0.5, pch = 16)
points(data$SOI, predict(PoiFull, type = "response"), col = "blue", cex = 0.5, pch = 16)
```



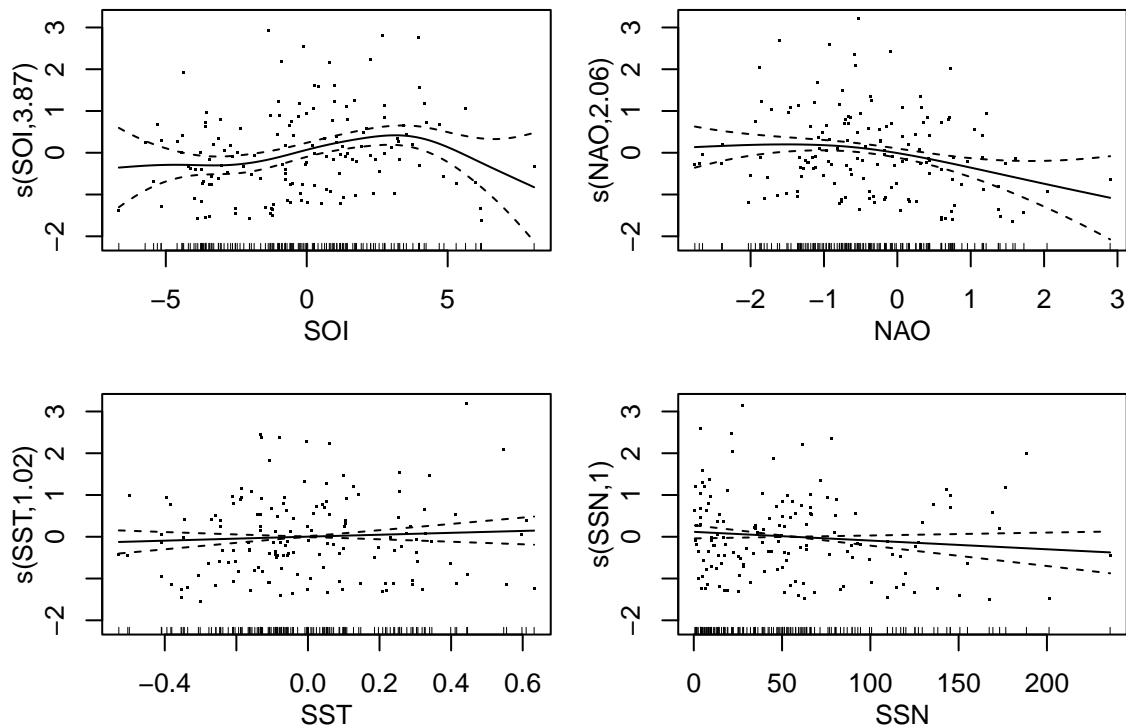
### Generalized additive Poisson regression

```
poi_gam1 <- gam(All ~ s(SOI) + s(NAO) + s(SST) + s(SSN), family = "poisson", data = data)
summary(poi_gam1)
```

```
##
## Family: poisson
## Link function: log
##
## Formula:
## All ~ s(SOI) + s(NAO) + s(SST) + s(SSN)
##
## Parametric coefficients:
## Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept) 0.48298 0.06721 7.187 6.64e-13 ***
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##      edf Ref.df Chi.sq p-value
## s(SOI) 3.875 4.855 18.975 0.00177 **
## s(NAO) 2.060 2.631 10.994 0.01019 *
## s(SST) 1.017 1.034 0.773 0.38045
## s(SSN) 1.000 1.000 2.245 0.13404
## ---
## Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) = 0.199 Deviance explained = 20.6%
## UBRE = 0.20739 Scale est. = 1 n = 145
```

```
par(mfrow = c(2, 2), mar = c(3.5, 3.5, 1, 0.5), mgp = c(2, 1, 0))
plot(poi_gam1, residuals = T)
```



```
par(mfrow = c(1, 1), mar = c(3.5, 3.5, 2, 2.5), mgp = c(2, 1, 0))
poi_gam2 <- gam(All ~ s(SOI, NAO, SST, SSN), family = "poisson", data = data)
summary(poi_gam2)
```

```
##
## Family: poisson
## Link function: log
##
## Formula:
## All ~ s(SOI, NAO, SST, SSN)
```

```

##
## Parametric coefficients:
##           Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.47320    0.06849   6.909 4.89e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##           edf Ref.df Chi.sq p-value
## s(SOI,NAO,SST,SSN) 14.31  14.6  32.39 0.00464 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) = 0.125  Deviance explained = 18.6%
## UBRE = 0.32248  Scale est. = 1          n = 145

```

```
plot(poi_gam2)
```

