Logistic Regression on the Plasma Data

Karen Mazidi

Logistic regression example using the plasma data set in package HSAUR.

Data exploration

We can read more about the plasma data set by typing "?plasma" at the console, after package HSAUR is loaded. We want to learn to predict ESR>20 or not, based on the levels of the plasma proteins fibrinogen and globulin. ESR stands for erythrocyte sedimentation rate, the rate at which red blood cells settle in blood plasma. Values >20 indicate some possible associations with various health conditions.

```
library(HSAUR)
```

```
## Loading required package: tools
```

```
attach(plasma)
str(plasma)
```

```
## 'data.frame':    32 obs. of  3 variables:
##  $ fibrinogen: num  2.52 2.56 2.19 2.18 3.41 2.46 3.22 2.21 3.15 2.6 ...
##  $ globulin  : int  38 31 33 31 37 36 38 37 39 41 ...
##  $ ESR       : Factor w/ 2 levels "ESR < 20","ESR > 20": 1 1 1 1 1 1 1 1 1 1 ...
```

```
head(plasma)
```

```
##   fibrinogen globulin      ESR
## 1       2.52       38 ESR < 20
## 2       2.56       31 ESR < 20
## 3       2.19       33 ESR < 20
## 4       2.18       31 ESR < 20
## 5       3.41       37 ESR < 20
## 6       2.46       36 ESR < 20
```

```
attach(plasma)
```

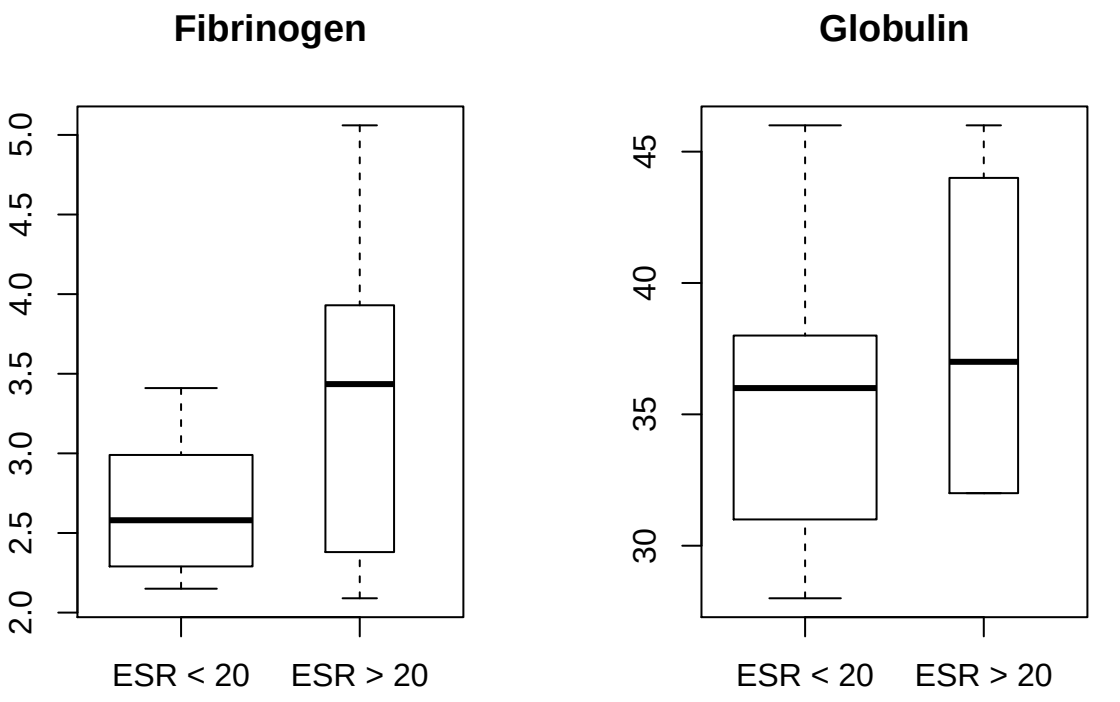
```
## The following objects are masked from plasma (pos = 3):
##
##     ESR, fibrinogen, globulin
```

Plot the data

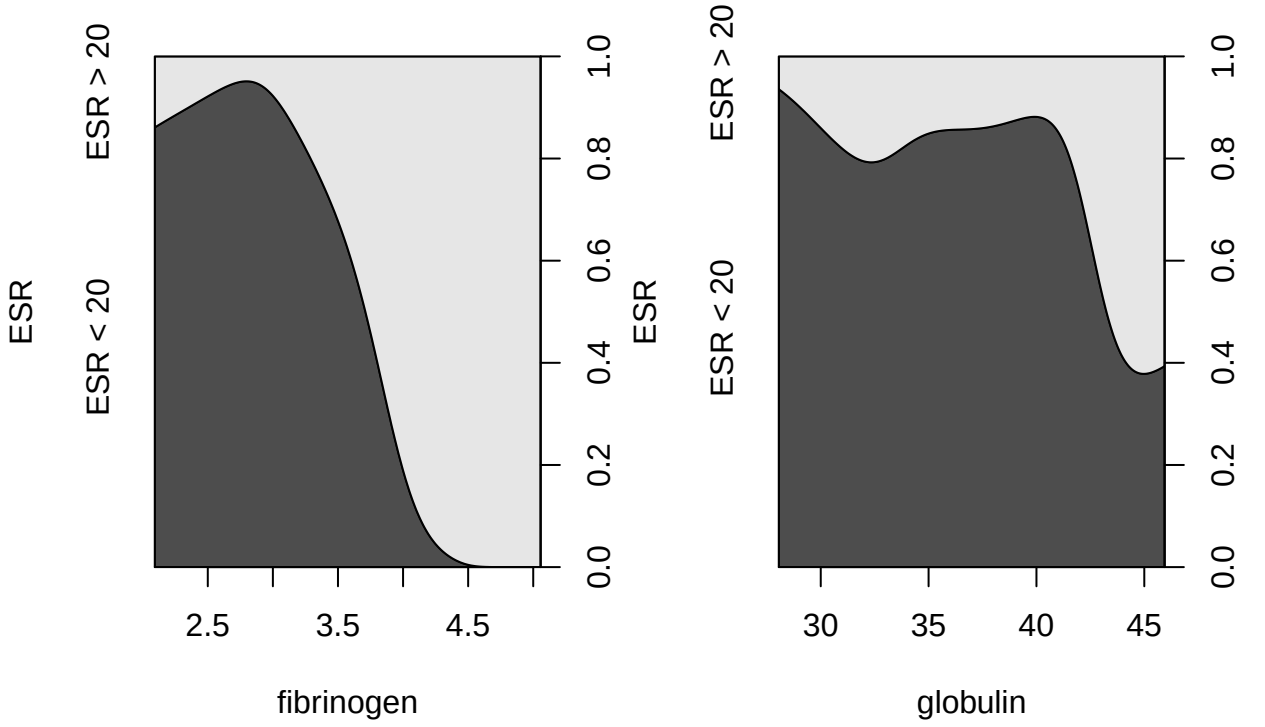
The first pair of plots show us that observations where ESR>20 are rarer. This is indicated by the thinner boxes because we set varwidth=TRUE in the bloxplot call. More importantly, the boxplots show that ESR>20 observations are associated with slightly higher levels of globulin and significantly higher levels of fibronogen.

The second set of pots are conditional density plots. We can make the same observations as the box plots. Here they are just visualized differently. The total probability space is the rectangle, with the lighter grey indicating ESR>20.

```
par(mfrow=c(1,2))
plot(ESR, fibrinogen, main="Fibrinogen", varwidth=TRUE)
plot(ESR, globulin, main="Globulin", varwidth=TRUE)
```




```
par(mfrow=c(1,2))
cdplot(ESR~fibrinogen)
cdplot(ESR~globulin)
```



Train and test sets

Even though our data is small, we will go ahead and divide it into train and test sets.

```
set.seed(1234)
i <- sample(1:nrow(plasma), 0.75*nrow(plasma), replace=FALSE)
```

```
train <- plasma[i,]
test <- plasma[-i,]
```

Build a logistic regression model

Our first model uses only fibronogen as a predictor. On our small test data we got about 88% accuracy. The table shows that all test observations were predicted as not ESR>20 and one of the 8 observations actually was ESR>20. Internally, the ESR>20 factor is coded as 1 for not >20 and 2 for ESR>20. This is why we compare them as.integer().

```
glm1 <- glm(ESR~fibrinogen, data=train, family=binomial)
summary(glm1)
```

```
##
## Call:
## glm(formula = ESR ~ fibrinogen, family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0112  -0.4648  -0.3517  -0.2692   2.6543
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -8.383      3.627  -2.311   0.0208 *
## fibrinogen     2.340      1.151   2.033   0.0421 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 24.564  on 23  degrees of freedom
## Residual deviance: 17.107  on 22  degrees of freedom
## AIC: 21.107
##
## Number of Fisher Scoring iterations: 5
```

```
probs <- predict(glm1, newdata=test, type="response")
pred <- ifelse(probs>0.5, 2, 1)
acc1 <- mean(pred==as.integer(test$ESR))
print(paste("glm1 accuracy = ", acc1))
```

```
## [1] "glm1 accuracy =  0.875"
```

```
table(pred, as.integer(test$ESR))
```

```
##
## pred 1 2
##    1 7 1
```

What does it mean?

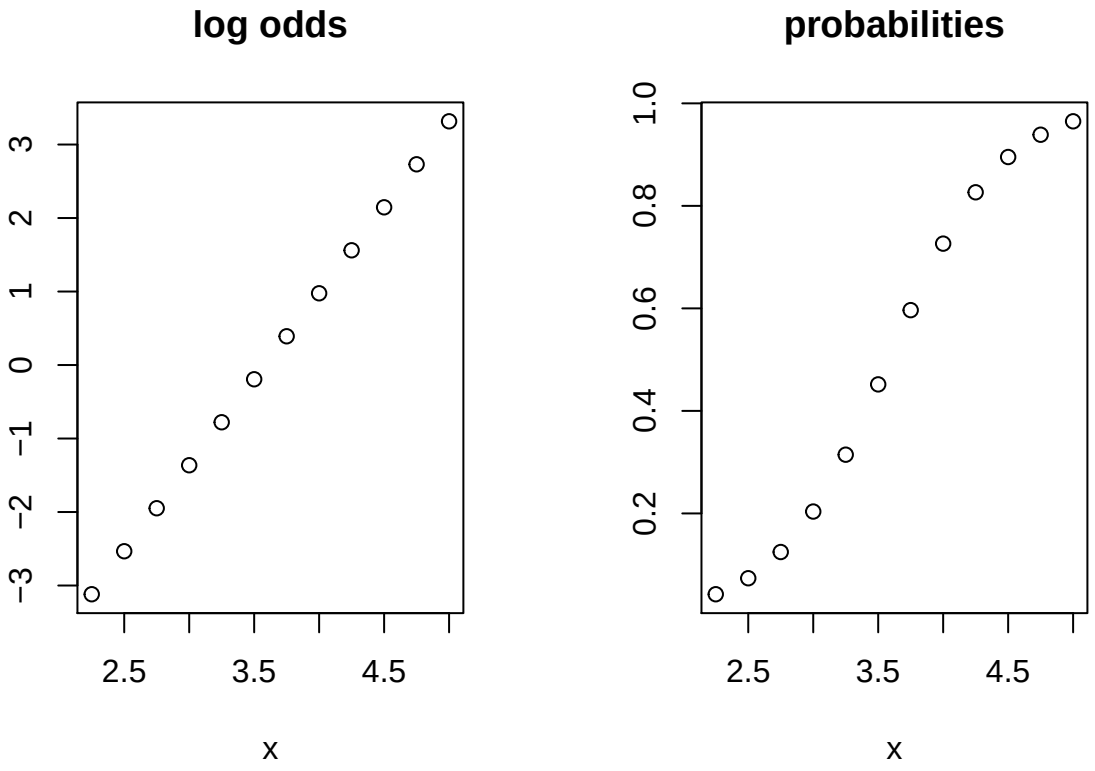
Let's explore the meaning of the coefficient.

```
fibro <- glm1$coefficients[2]
intercept <- glm1$coefficients[1]

log_odds <- function(x, fibro, intercept){
  intercept + fibro * x
}

x <- seq(from=2.25, to=5.0, by=0.25)
y <- log_odds(x, fibro, intercept)
par(mfrow=c(1,2))
plot(x,y, main="log odds", ylab="")

prob <- exp(y) / (1 + exp(y))
plot(x, prob, main="probabilities", ylab="")
```



Build another model

This model uses both predictors. On the test set we got the same accuracy.

```
glm2 <- glm(ESR~fibrinogen+globulin, data=train, family=binomial)
summary(glm2)
```

```
##
## Call:
## glm(formula = ESR ~ fibrinogen + globulin, family = binomial,
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
```

```
## -1.1000  -0.5644  -0.2436  -0.1374   2.1894
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -16.5677     8.1851  -2.024   0.0430 *
## fibrinogen    2.6543     1.3800   1.923   0.0544 .
## globulin      0.1982     0.1476   1.343   0.1794
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 24.564  on 23  degrees of freedom
## Residual deviance: 14.892  on 21  degrees of freedom
## AIC: 20.892
##
## Number of Fisher Scoring iterations: 6
```

```
probs <- predict(glm1, newdata=test, type="response")
pred <- ifelse(probs>0.5, 2, 1)
acc2 <- mean(pred==as.integer(test$ESR))
print(paste("glm2 accuracy = ", acc2))
```

```
## [1] "glm2 accuracy =  0.875"
```

```
table(pred, as.integer(test$ESR))
```

```
##
## pred 1 2
##    1 7 1
```

Compare the models with anova()

The second model is only slightly better than the first. The residuals dropped by only 2 points, and the p-value is not small.

```
anova(glm1, glm2)
```

```
## Analysis of Deviance Table
##
## Model 1: ESR ~ fibrinogen
## Model 2: ESR ~ fibrinogen + globulin
##   Resid. Df Resid. Dev Df Deviance
## 1        22     17.107
## 2        21     14.892  1   2.2146
```