AdvStDaAn, Worksheet, Week 4

Micheal Lappert

21.04.2022

Contents

Exercise 1 . . . 1
- Exercise 1.a) . . . 3
- Exercise 1.b) . . . 5
- Exercise 1.c) . . . 6
- Question 1.c) . . . 6
- Exercise 1.d) . . . 6
- Exercise 1.e) . . . 6
- Exercise 1.f) . . . 7

Exercise 2 . . . 8
- Exercise 2.a) . . . 10
- Question 2.a) . . . 10
- Exercise 2.b) . . . 12
- Exercise 2.c) . . . 14

Exercise 1

```
path <- file.path('Datasets', 'turbines.dat')
df <- read.table(path, header=TRUE)

summary(df)
```

Dataset loading and sanity check:

```
##      Hours        Turbines       Fissures
##  Min.   : 400   Min.   :13.00   Min.   : 0.000
##  1st Qu.:1600   1st Qu.:33.50   1st Qu.: 4.500
##  Median :2600   Median :39.00   Median : 7.000
##  Mean   :2582   Mean   :39.27   Mean   : 9.636
##  3rd Qu.:3600   3rd Qu.:41.00   3rd Qu.:15.000
##  Max.   :4600   Max.   :73.00   Max.   :22.000
```

```r
str(df)
```

```
## 'data.frame':    11 obs. of  3 variables:
##  $ Hours   : int  400 1000 1400 1800 2200 2600 3000 3400 3800 4200 ...
##  $ Turbines: int  39 53 33 73 30 39 42 13 34 40 ...
##  $ Fissures: int  0 4 2 7 5 9 9 6 22 21 ...
```

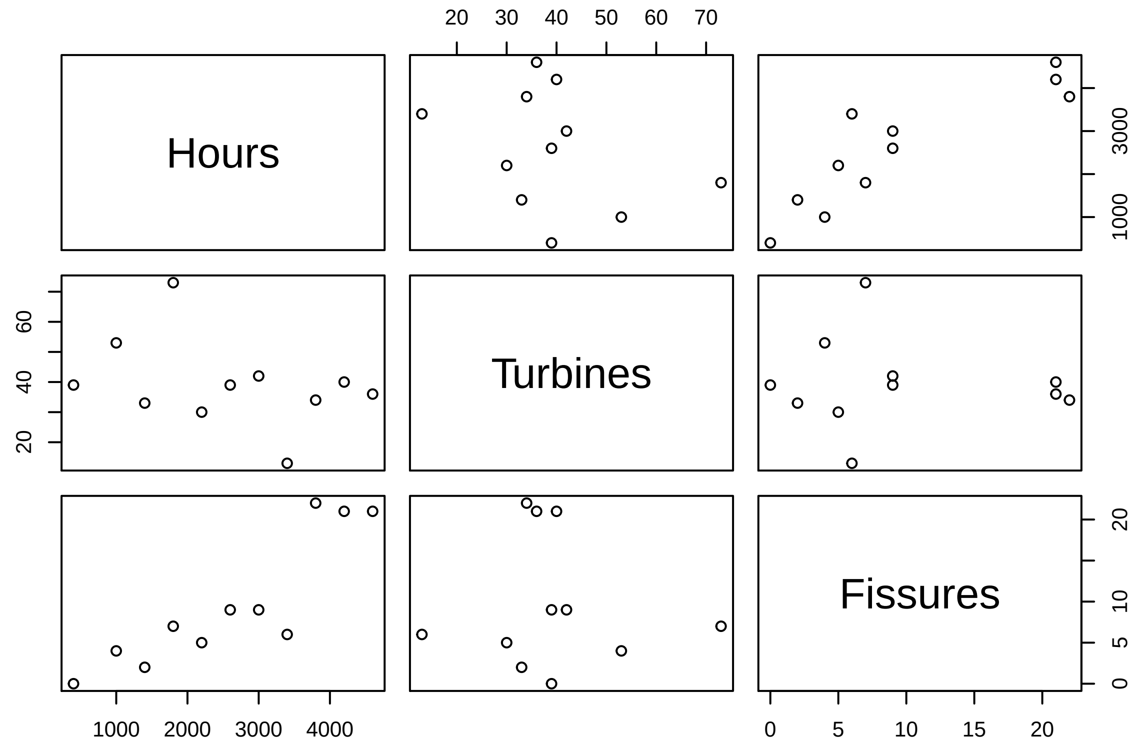
```r
head(df)
```

```
##   Hours Turbines Fissures
## 1   400       39        0
## 2  1000       53        4
## 3  1400       33        2
## 4  1800       73        7
## 5  2200       30        5
## 6  2600       39        9
```

```r
tail(df)
```

```
##    Hours Turbines Fissures
## 6   2600       39        9
## 7   3000       42        9
## 8   3400       13        6
## 9   3800       34       22
## 10  4200       40       21
## 11  4600       36       21
```

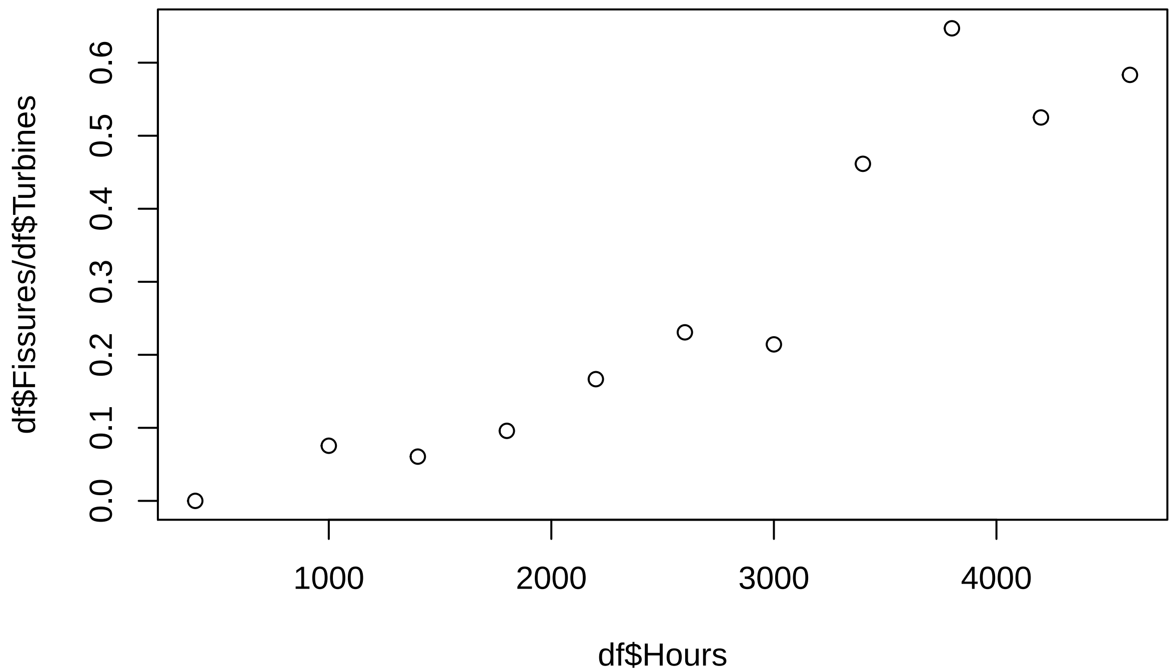
```r
plot(df)
```



The data is ascending sorted in hours and looks fine.

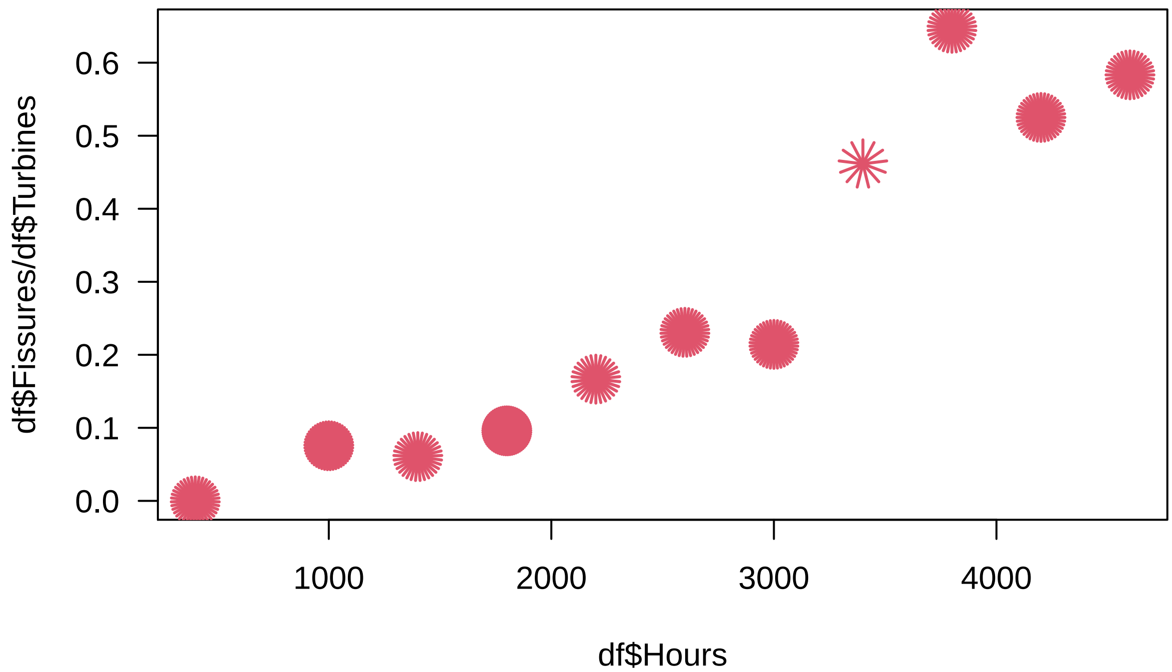
Exercise 1.a)

```
par(mfrow=c(1,1))
plot(df$Hours, df$Fissures/df$Turbines)
```



This plot does not show the density per observation. So one might consider an alternative plot where the densitiy is visualized as well.

```
sunflowerplot(df$Hours, df$Fissures/df$Turbines,
              number = df$Turbines, las = 1)
```



The sunflowerplot is better suited for this purpose: The more 'lines' are at one postiion, the more observations are there.

Exercise 1.b)

Let Y_i be the number of wheels with fissures. Then
$Y_i \sim$ independent Binomial(π_i, #Turbines)
with
$log(\frac{\pi_i}{1-\pi_i}) = \beta_0 + \beta_1 * Hours$

```
glm1.1 <- glm(cbind(Fissures, Turbines-Fissures) ~ Hours, family = binomial, data = df)
summary(glm1.1)
```

```
##
## Call:
## glm(formula = cbind(Fissures, Turbines - Fissures) ~ Hours, family = binomial,
##     data = df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5055  -0.7647  -0.3036   0.4901   2.0943
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.9235966  0.3779589 -10.381   <2e-16 ***
```

```
## Hours        0.0009992  0.0001142   8.754   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 112.670  on 10  degrees of freedom
## Residual deviance:  10.331  on  9  degrees of freedom
## AIC: 49.808
##
## Number of Fisher Scoring iterations: 4
```

Exercise 1.c)

```
coef(glm1.1)
```

```
##   (Intercept)         Hours
## -3.9235965551  0.0009992372
```

$log(\frac{\pi_i}{1-\pi_i}) = -3.9235965551 + 0.0009992372 * Hours$

Hence the probability of fissures increases by a facotr of exp(0.0009992372) = 1.0009997

Question 1.c)

How do we know that the increase of the probability of fissures is related to 100 hours? Why not per 1 hour?

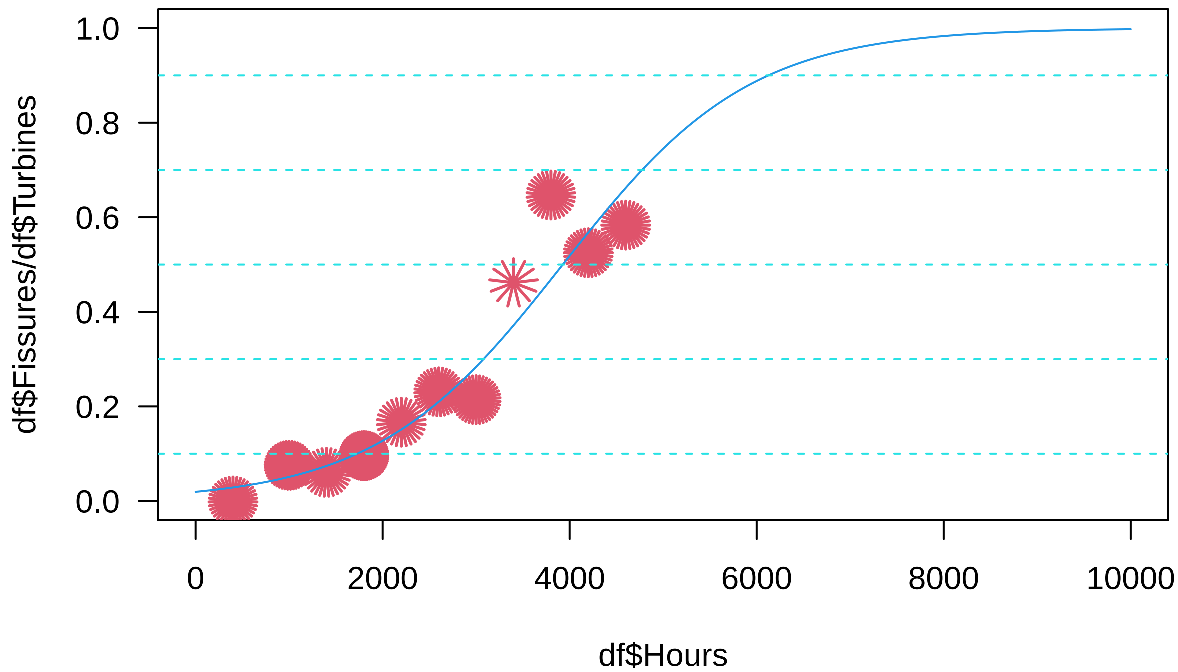
Exercise 1.d)

```
preds1 <- predict(glm1.1, type = "response", newdata = data.frame(Hours = 3000))
```

The estimated probability of a 'defective' turbine wheel with operation time 3'000 is 0.2837603

Exercise 1.e)

```
sunflowerplot(df$Hours, df$Fissures/df$Turbines,
              number = df$Turbines, las = 1,
              ylim = c(0, 1), xlim = c(0, 10000))
dfPred <- data.frame(Hours = seq(from = 0, to = 10000, by = 10))
preds2 <- predict(glm1.1, type = 'response', newdata = dfPred)
lines(dfPred$Hours, preds2, col = 4)
abline(h = c(0.1, 0.3, 0.5, 0.7, 0.9), lty = 2, col = 5)
```



Exercise 1.f)

Fitting the logistic regression model with the probit and the cloglog link functions.

```
glm1.probit <- glm(cbind(Fissures, Turbines-Fissures) ~ Hours,
                   family = binomial(link = probit),
                   data = df)

glm1.cloglog <- glm(cbind(Fissures, Turbines-Fissures) ~ Hours,
                    family = binomial(link = cloglog),
                    data = df)
```

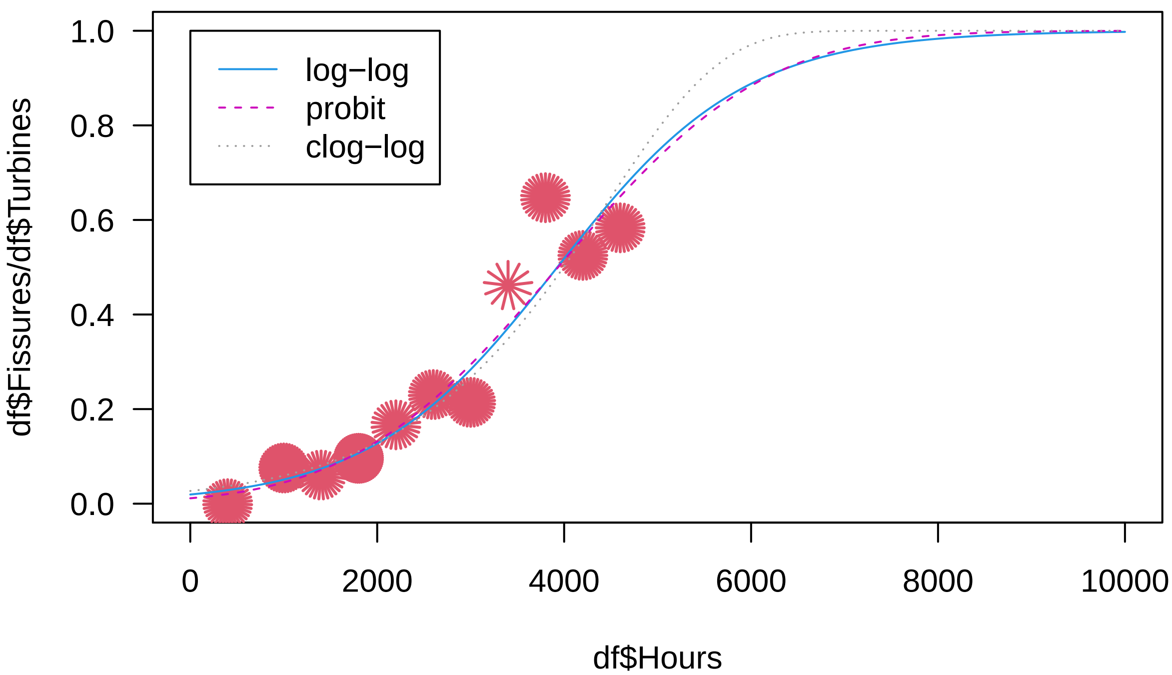
The coefficients of the models are:
log-log: -3.9235966, 9.9923723×10^{-4}
probit: -2.2758075, 5.7832109×10^{-4}
cloglog: -3.6032798, 8.1049362×10^{-4}
The coefficients of the log-log and the cloglog model are similar, whereas the probit models has lower coefficients.

Predicting and plotting the corresponding curves of the different models.

```
predsProbit <- predict(glm1.probit, type = 'response', newdata = dfPred)
predsCloglog <- predict(glm1.cloglog, type = 'response', newdata = dfPred)

sunflowerplot(df$Hours, df$Fissures/df$Turbines,
```

```
             number = df$Turbines, las = 1,
             ylim = c(0, 1), xlim = c(0, 10000))
lines(dfPred$Hours, preds2, col = 4)
lines(dfPred$Hours, predsProbit, col = 6, lty = 2)
lines(dfPred$Hours, predsCloglog, col = 8, lty = 3)
legend(x = 1, y = 1, legend = c('log-log', 'probit', 'clog-log'),
       col = c(4, 6, 8),
       lty = c(1, 2, 3))
```



Looking at the plot the before stated picture changes: The log-log and the probit model look more similar than the corresponding clog-log model. -> When comparing models, rather look at the corresponding curves than the coefficients!

Exercise 2

```
path <- file.path('Datasets', 'birth-weight.dat')
df <- read.table(path, header = TRUE)

str(df)
```

Dataset loading and sanity check:

```
## 'data.frame':	10 obs. of  3 variables:
##  $ m     : int  10 14 27 22 32 28 22 26 34 32
##  $ Y     : int  0 2 9 8 23 21 19 19 31 29
##  $ weight: int  550 650 750 850 950 1050 1150 1250 1350 1450
```

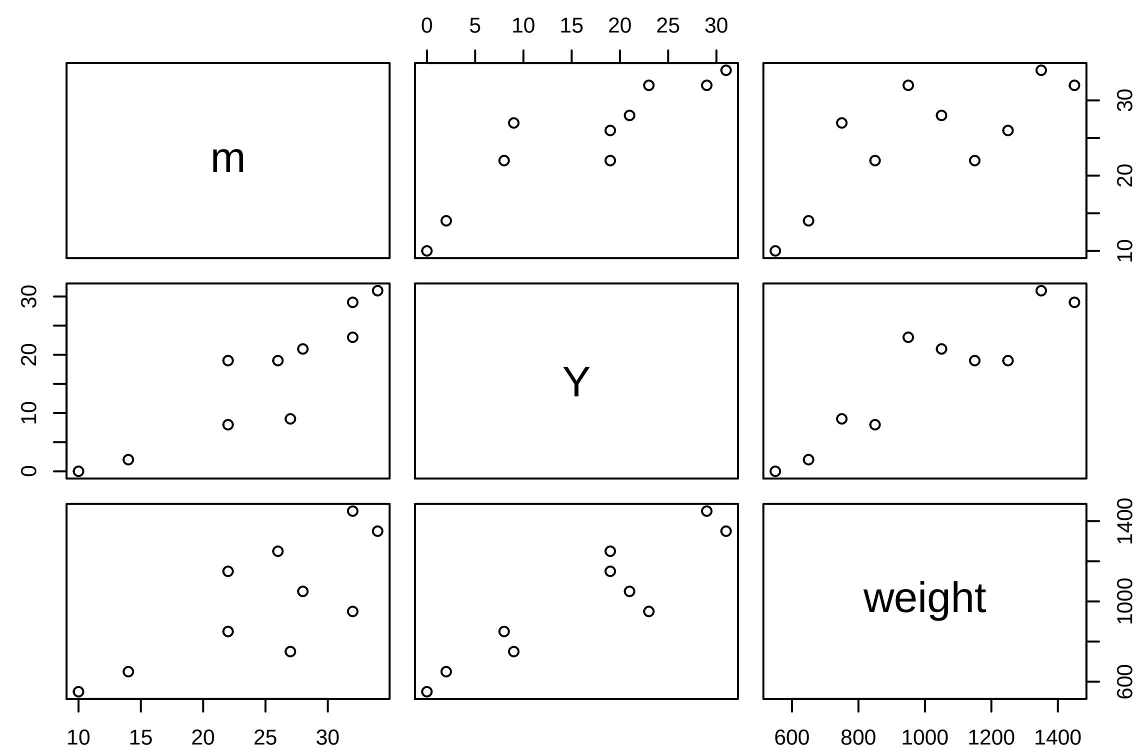
```
head(df)
```

```
##    m  Y weight
## 1 10  0    550
## 2 14  2    650
## 3 27  9    750
## 4 22  8    850
## 5 32 23    950
## 6 28 21   1050
```

```
tail(df)
```

```
##     m  Y weight
## 5  32 23    950
## 6  28 21   1050
## 7  22 19   1150
## 8  26 19   1250
## 9  34 31   1350
## 10 32 29   1450
```

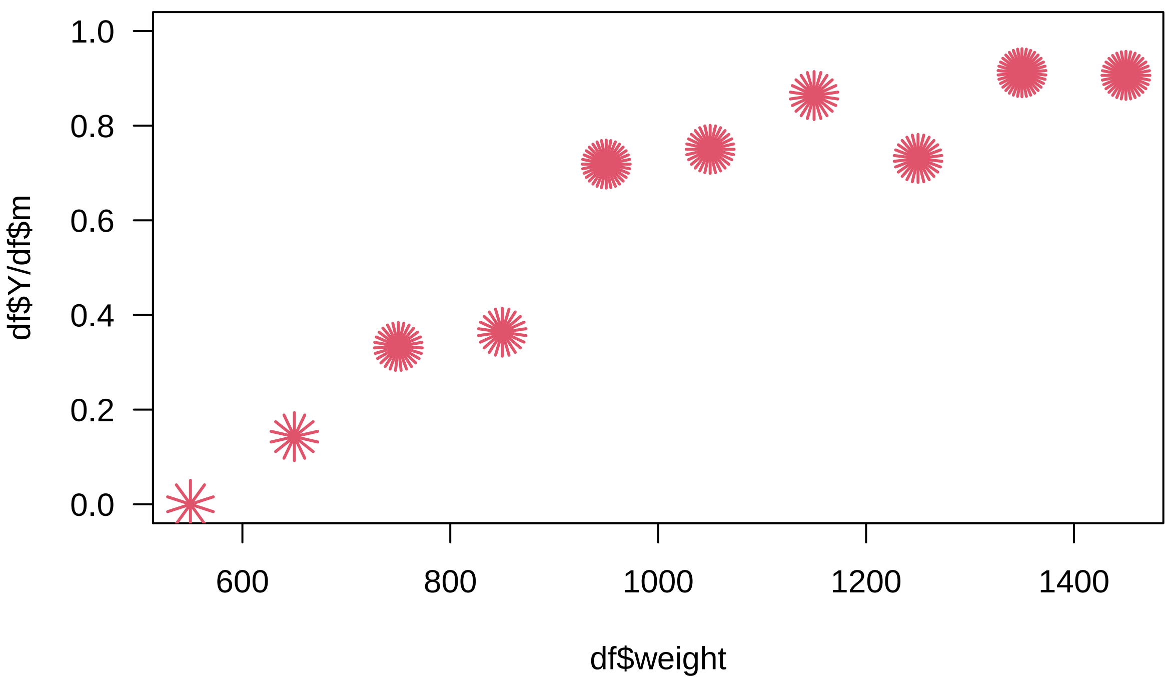
```
plot(df)
```



The data seems to be sorted in weight and also some correlation between Y and the explanatory variables seems obvious.

Exercise 2.a)

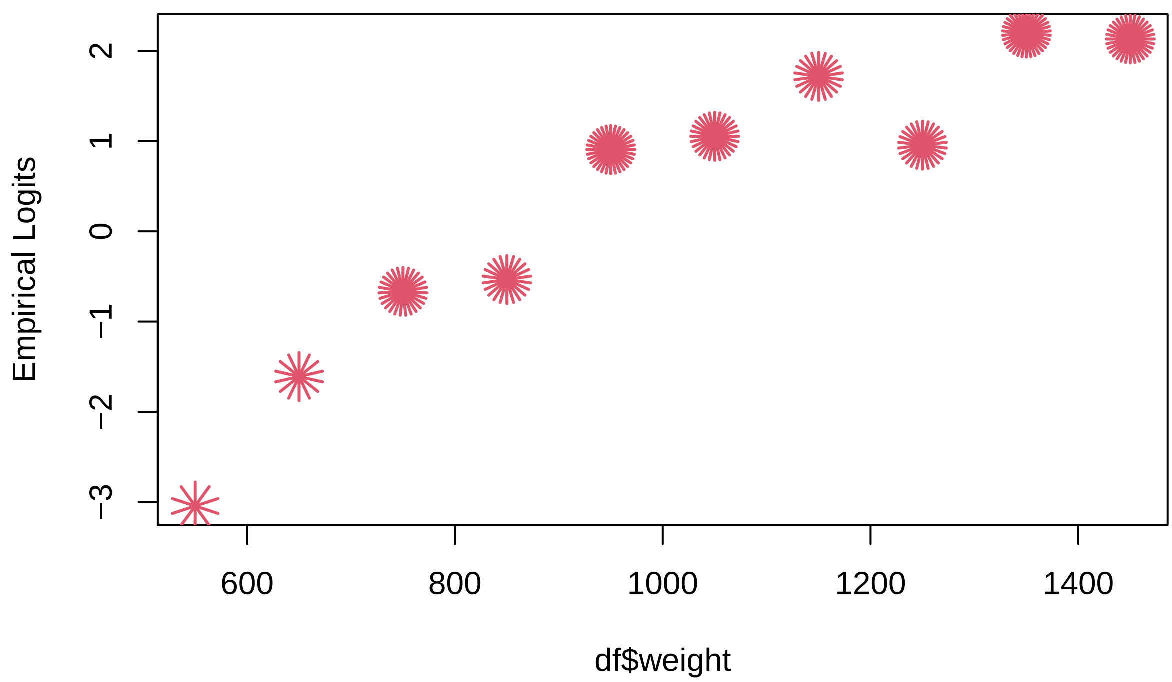
```
sunflowerplot(df$weight, df$Y/df$m,
              number = df$m,
              las = 1, ylim = c(0,1))
```



Question 2.a)

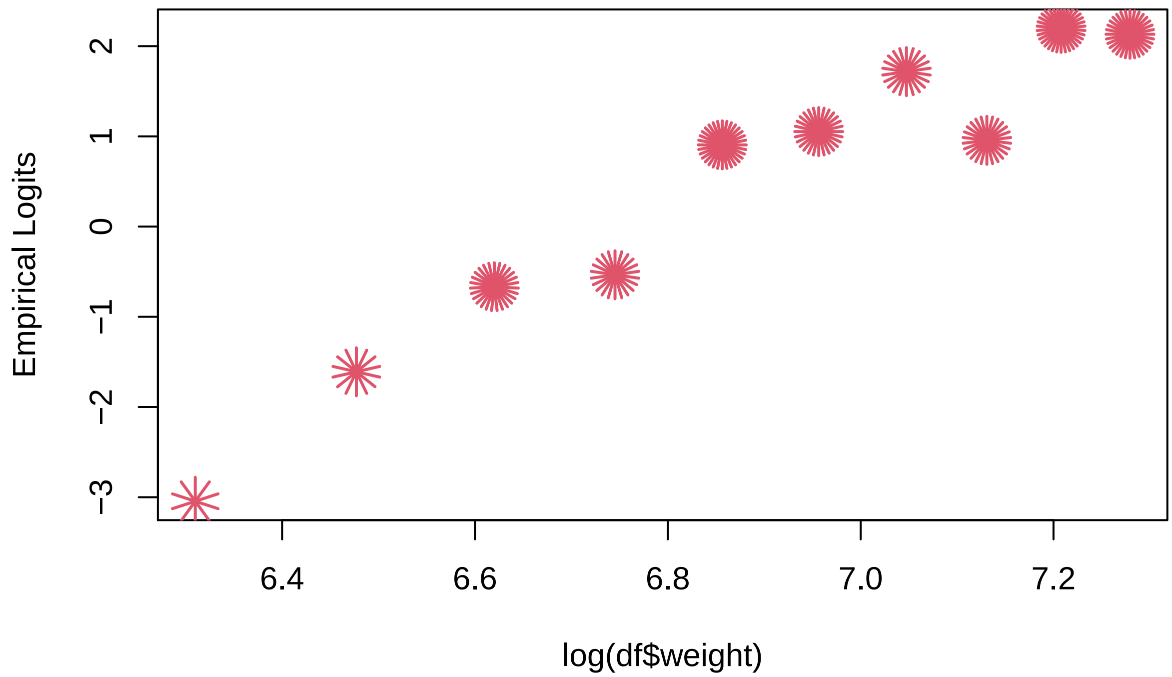
Why is for the calculation of the empirical logits the formula like this? Why not only: log((Y+0.5)/(m)) -> And why is the +0.5?

```
yel <- log((df$Y+0.5)/(df$m-df$Y+0.5))
sunflowerplot(x=df$weight, y=yel, number=df$m, ylab="Empirical Logits")
```



This does not look like a linear relationship. So one would try an additional transformation.

```
sunflowerplot(x=log(df$weight), y=yel, number=df$m, ylab="Empirical Logits")
```



This looks much better. So we transform weight in the dataset

```
df$lWeight <- log(df$weight)
```

Exercise 2.b)

Fitting the model using the least squares approach with empirical logits.

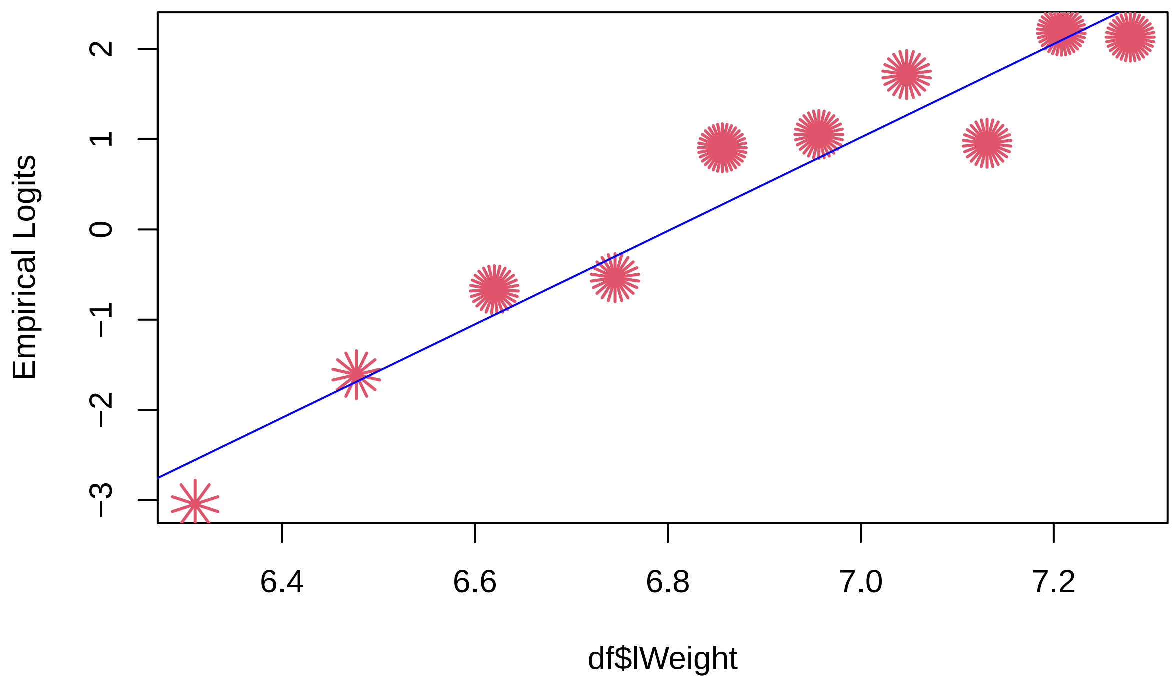
```
lm2.1 <- lm(yel ~ lWeight, data = df)
summary(lm2.1)
```

```
##
## Call:
## lm(formula = yel ~ lWeight, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.74210 -0.30957  0.09027  0.27564  0.62935
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -35.2318     3.2958  -10.69 5.15e-06 ***
## lWeight       5.1788     0.4797   10.79 4.78e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.4638 on 8 degrees of freedom
## Multiple R-squared:  0.9358, Adjusted R-squared:  0.9277
## F-statistic: 116.5 on 1 and 8 DF,  p-value: 4.782e-06
```

Plotting the resulting regression line in the previous plot.

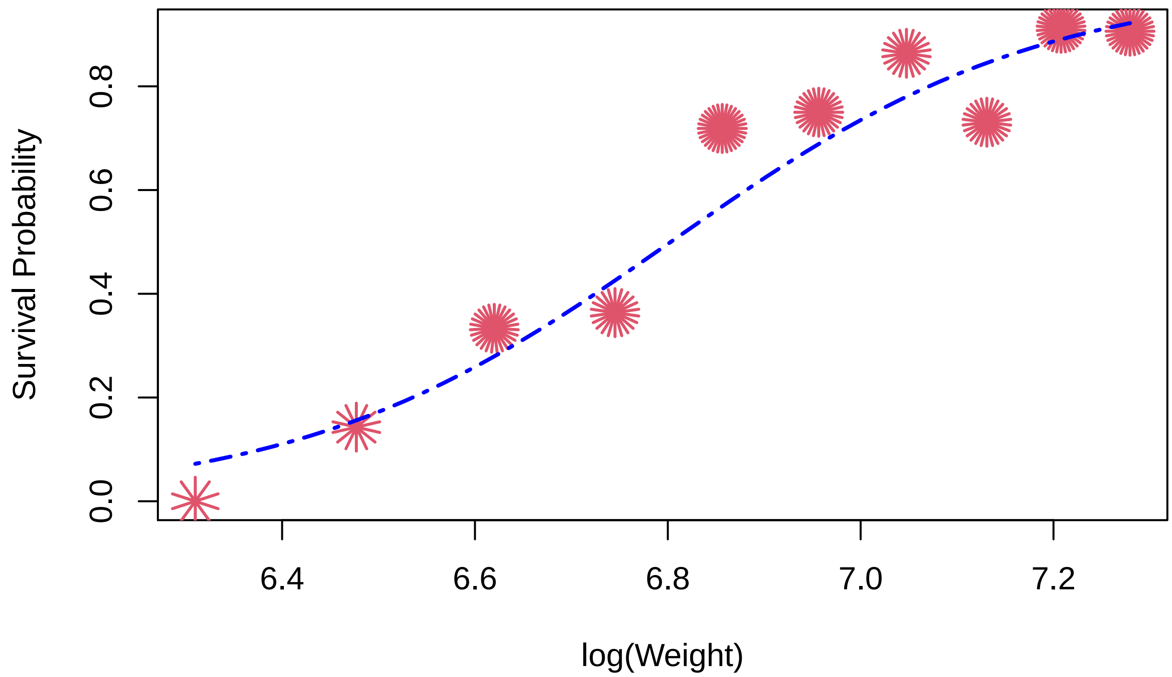
```
sunflowerplot(x=df$lWeight, y=yel, number=df$m, ylab="Empirical Logits")
abline(lm2.1, col = 'blue')
```



Display data and fit in the response scale

```
sunflowerplot(x=df$lWeight, y=df$Y/df$m, number=df$m,
              xlab="log(Weight)", ylab="Survival Probability")
x <- seq(min(df$lWeight), max(df$lWeight), length=50)
mu.logit.p <- predict(lm2.1, newdata=data.frame(lWeight=x))

# Back-transformation into the response scale:
lines(x, 1/(1+exp(-mu.logit.p)), col="blue", lwd=2, lty=6)
```



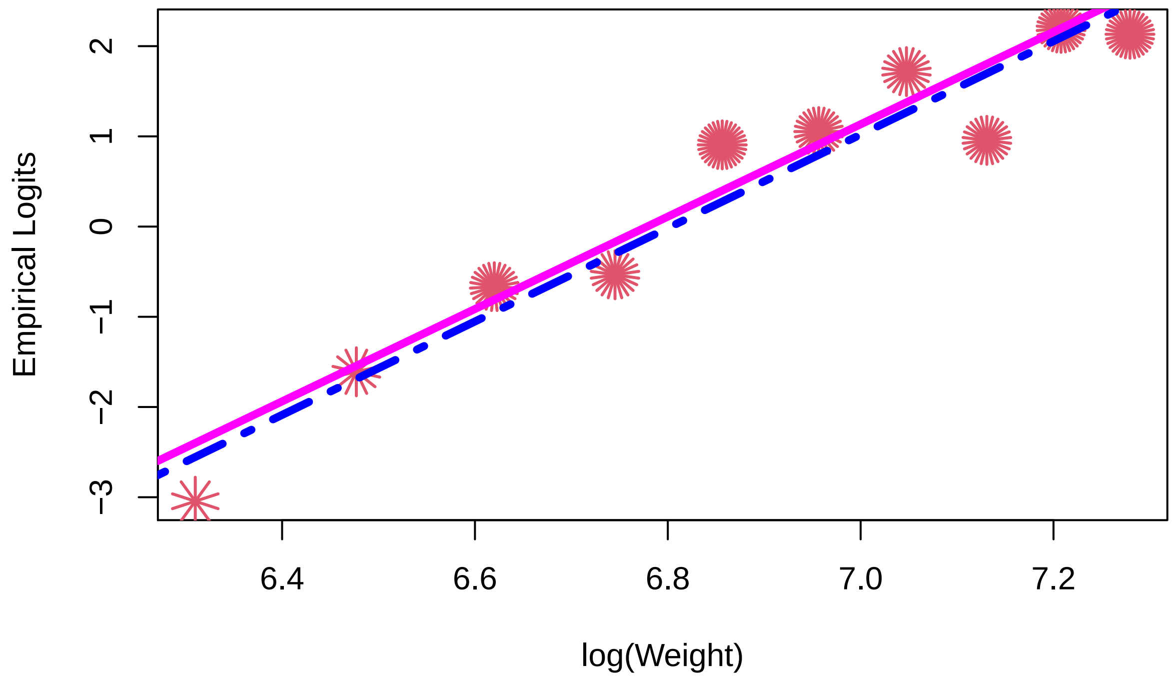
Exercise 2.c)

Fitting the glm:

```
glm2.1 <- glm(cbind(Y, m-Y) ~ lWeight, family = binomial, data = df)
```

Display data and both fits in the logitic scale:

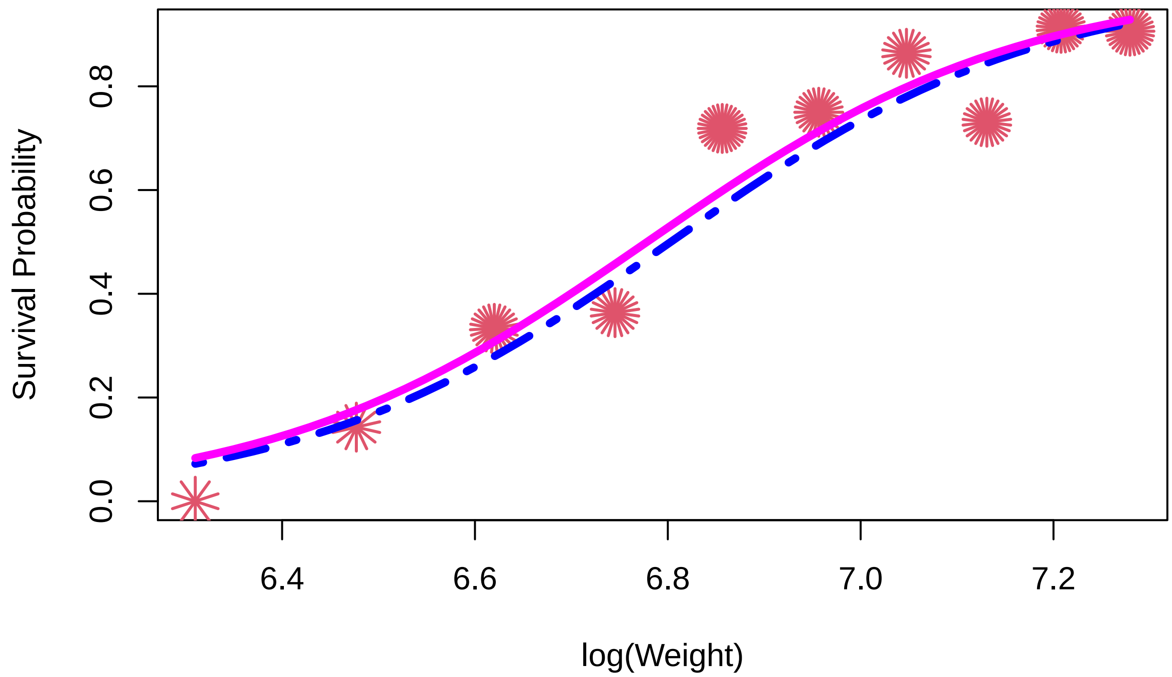
```
sunflowerplot(x=df$lWeight, y=yel, number=df$m,
              xlab="log(Weight)", ylab="Empirical Logits")
abline(lm2.1, col="blue", lwd=4, lty=6)
abline(coef(glm2.1), col="magenta", lwd=4) ## glm fit
```



Two almost paralell lines.

Display data and both fits in the response scale:

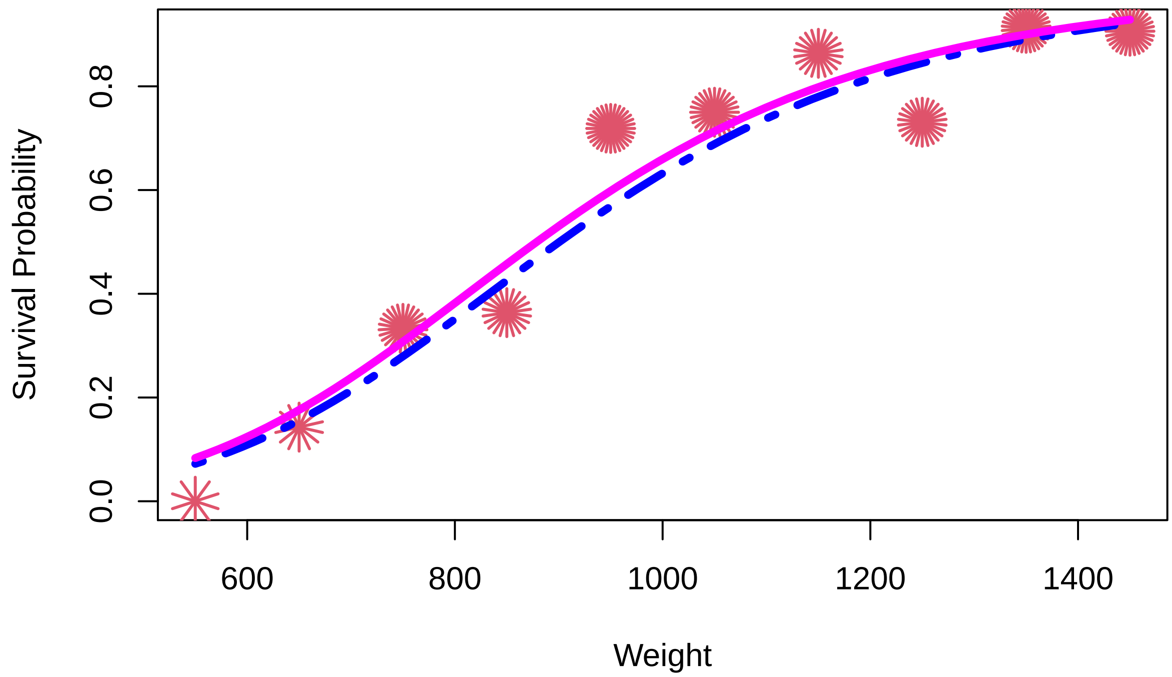
```
sunflowerplot(x=df$lWeight, y=df$Y/df$m, number=df$m,
              xlab="log(Weight)", ylab="Survival Probability")
x <- seq(min(df$lWeight), max(df$lWeight), length=50)
mu.logit.p <- predict(lm2.1, newdata=data.frame(lWeight=x))
lines(x, 1/(1+exp(-mu.logit.p)), col="blue", lwd=4, lty=6)
mu.glm.p <- predict(glm2.1, newdata=data.frame(lWeight=x), type="response")
lines(x, mu.glm.p, col="magenta", lwd=4)
```



Difference in the fits is small. It might be that the glm fit fits better on the r.h.s.

Display data and both fits in a scatterplot of the response against Weights:

```
sunflowerplot(x=df$weight, y=df$Y/df$m, number=df$m,
              xlab="Weight", ylab="Survival Probability")
lines(exp(x), 1/(1+exp(-mu.logit.p)), col='blue', lwd=4, lty=6)
lines(exp(x), mu.glm.p, col="magenta", lwd=4) ## from GLM
```


Survival Probability
0.0
0.2
0.4
0.6
0.8
600
800
1000
1200
1400
Weight