Statistical Methods

Laboratory 4

N Torelli, G Di Credico, V Gioia

27 - 11 - 2023

Contents

Generalized Linear Models

Many times, linearity between a dependent variable and a set of predictors is not a suited assumption. Normal linear models rely on a *normality assumption*, and this may be a limitation when the nature of the dependent variable is discrete, e.g. with support $\{0, 1\}$ or $\{0, 1, 2, \dots, \}$, or it is continuous and its range spans from 0 to $+\infty$.

In normal linear models, the main assumption is:

$$E[Y_i|x_i] = \eta_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}.$$

while the generalized linear model consider modelling by using covariates a suitable transformation of the conditional expectation for a broad range of distributions, that is

$$g(E[Y_i|x_i]) = \eta_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}.$$

The main ingredients of a GLM are: the distibution of Y_i, which belongs to the exponential dispersion family, the linear predictor, and the **link function** g that should be twice differentiable.

Logistic regression

The following example is from: *Data analysis using regression and multilevel/ hierarchical models*, Gelman, A., & Hill, J. (2007), Cambridge University Press.

It is known that water in some wells in Bangladesh and other South Asian countries might be contaminated with natural arsenic. The effect of the arsenic ingestion on health is a cumulative poisoning, and the risk of cancer and other diseases is estimated to be proportional to the dose intake.

A research team from the United States and Bangladesh examined the water of several wells in Araihazar district of Bangladesh and they measured the arsenic level classifying the wells as *safe* if the arsenic level was below the 0.5 in units of hundreds of micrograms per liter, or *unsafe* if it was above 0.5. All the people using unsafe wells have been encouraged to switch to a nearby well. Indeed the contamination does not depend on the proximity and close wells can have very different levels of arsenic.

The dataset in the file `Wells.csv` contains information on whether or not 3020 households changed the wells that they were using after few years from the arsenic investigation. The variables are

- **switch:**: whether or not the household switched to another well from an unsafe well: no (0) or yes (1);
- **arsenic**: the level of arsenic contamination in the household's original well (in hundreds of micrograms per liter); note that all are above 0.5, which was the level identified as "unsafe";
- **distance**: distance to the closest known safe well (in meters);
- **education**: education of the head of the household (in years);
- **association:** whether or not any members of the household participated in any community organizations: no (0) or yes (1).

Goal: using a logistic regression, we would determine which are the main factors of well switching among the users of unsafe wells.

We start by reading the data, by analysing the structure and exploring some characteristics of the data.

```
data <- read.csv2("Wells.csv", header = TRUE)
str(data)
```

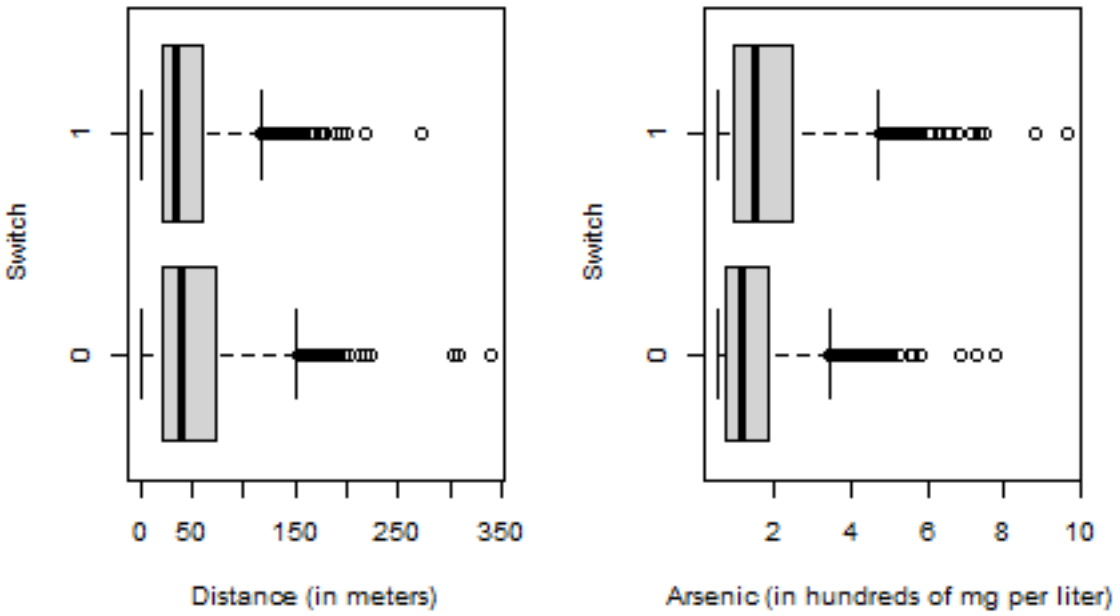
```
## 'data.frame':    3020 obs. of  5 variables:
##  $ switch : int  1 1 0 1 1 1 1 1 1 1 ...
##  $ arsenic: num  2.36 0.71 2.07 1.15 1.1 3.9 2.97 3.24 3.28 2.52 ...
##  $ dist   : num  16.8 47.3 21 21.5 40.9 ...
##  $ assoc  : int  0 0 0 0 1 1 1 0 1 1 ...
##  $ educ   : int  0 0 10 12 14 9 4 10 0 0 ...
```

```
summary(data)
```

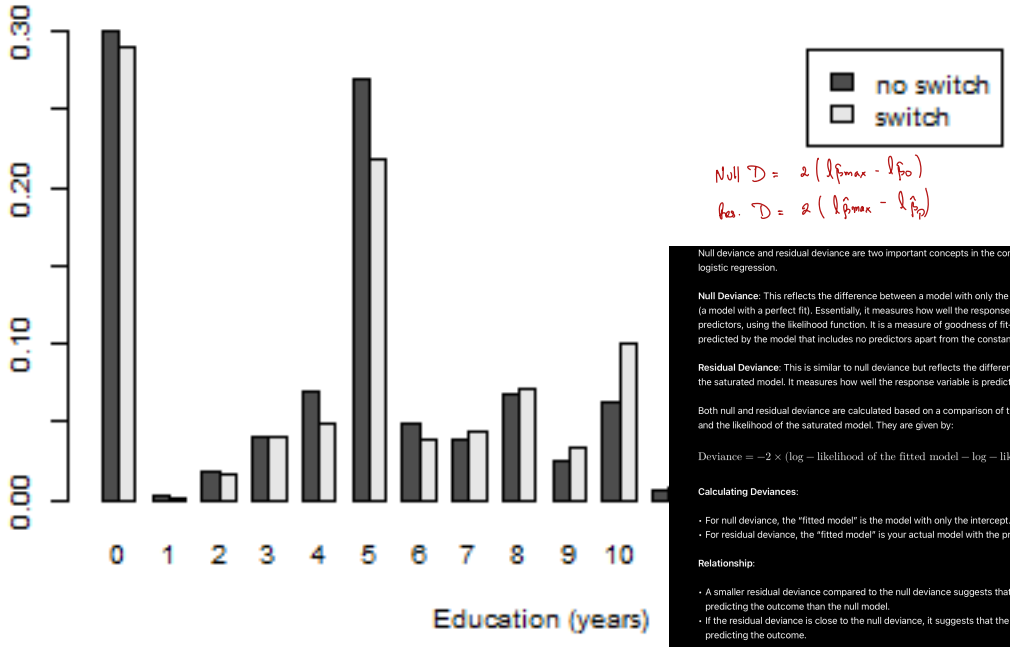
```
##      switch          arsenic           dist             assoc             educ      
##  Min.   :0.0000   Min.   :0.510   Min.   :  0.387   Min.   :0.0000   Min.   : 0.000  
##  1st Qu.:0.0000   1st Qu.:0.820   1st Qu.: 21.117   1st Qu.:0.0000   1st Qu.: 0.000  
##  Median :1.0000   Median :1.300   Median : 36.761   Median :0.0000   Median : 5.000  
##  Mean   :0.5752   Mean   :1.657   Mean   : 48.332   Mean   :0.4228   Mean   : 4.828  
##  3rd Qu.:1.0000   3rd Qu.:2.200   3rd Qu.: 64.041   3rd Qu.:1.0000   3rd Qu.: 8.000  
##  Max.   :1.0000   Max.   :9.650   Max.   :339.531   Max.   :1.0000   Max.   :17.000
```

In a very brief exploratory analysis, we analyse the distribution of distance, arsenic and education by switch

```
par(mfrow = c(1, 2))
boxplot(dist ~ switch, data = data, horizontal = TRUE,
        xlab = "Distance (in meters)", ylab = "Switch")
boxplot(arsenic ~ switch, data = data, horizontal = TRUE,
        xlab = "Arsenic (in hundreds of mg per liter)", ylab = "Switch")
```




```
par(mfrow = c(1, 1))
barplot(prop.table(table(data$switch, data$educ), margin = 1),
        beside = TRUE, legend.text = c("no switch", "switch"), xlab = "Education (years)")
```



$\text{Null } D = 2(l_{\hat{p}_{max}} - l_{\hat{p}_0})$

$\text{Res. } D = 2(l_{\hat{p}_{max}} - l_{\hat{p}_p})$

Null deviance and residual deviance are two important concepts in the context of Generalized Linear Models (GLMs), such as logistic regression.

Null Deviance: This reflects the difference between a model with only the intercept (no predictors) and the saturated model (a model with a perfect fit). Essentially, it measures how well the response variable is predicted by a model with no predictors, using the likelihood function. It is a measure of goodness of fit—specifically, it shows how well the outcome is predicted by the model that includes no predictors apart from the constant.

Residual Deviance: This is similar to null deviance but reflects the difference between the fitted model (with predictors) and the saturated model. It measures how well the response variable is predicted by the model with predictors included.

Both null and residual deviance are calculated based on a comparison of two likelihoods: the likelihood of the fitted model and the likelihood of the saturated model. They are given by:

$$\text{Deviance} = -2 \times (\log - \text{likelihood of the fitted model} - \log - \text{likelihood of the saturated model})$$

Calculating Deviances:

- For null deviance, the "fitted model" is the model with only the intercept.
- For residual deviance, the "fitted model" is your actual model with the predictors included.

Relationship:

- A smaller residual deviance compared to the null deviance suggests that the model with predictors is doing a better job of predicting the outcome than the null model.
- If the residual deviance is close to the null deviance, it suggests that the predictors are not adding much in terms of predicting the outcome.

Usage:

- These values are used to conduct a goodness-of-fit test for the model. In logistic regression, if the residual deviance is significantly lower than the null deviance, it suggests that the model fits the data well.
- They are also used in comparing nested models. The change in deviance (null deviance - residual deviance) follows a chi-square distribution and can be used to perform a likelihood ratio test, which can help determine if the predictors in the model significantly improve the fit.

Logistic regression with a single predictor

We start by modelling the effect of the variable `distance` on the variable `switch`. Since the outcome variable is a binary variable, the model chosen is the logistic regression model.

$$\Pr(Y_i = 1) = E(Y_i) = p_i \quad \text{logit}(p_i) = \mathbf{x}_i^\top \beta = \beta_0 + \beta_1 \text{dist}_i$$

where $\text{logit}(x) = \log(x/(1-x))$ is a function that maps the range $(0, 1)$ to the range $(-\infty, \infty)$.

```
fit1 <- glm(switch ~ dist , data = data,
            family = binomial(link = "logit"))
summary(fit1)
```

```
##
## Call:
## glm(formula = switch ~ dist, family = binomial(link = "logit"),
##     data = data)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.6059594  0.0603102  10.047  < 2e-16 ***
## dist        -0.0062188  0.0009743  -6.383 1.74e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 4076.2  on 3018  degrees of freedom
## AIC: 4080.2
##
## Number of Fisher Scoring iterations: 4
```

What is this?

Comments:

- The intercept can only be interpreted assuming zero values for the other predictors. Here, when `dist`$= 0$ he probability of switching well for a family who lives close to a safe well is about 65%, given by

$$P(Y_i = 1 | dist_i = 0) = \frac{e^{\beta_0}}{1 + e^{\beta_0}} = \frac{1}{1 + e^{-\beta_0}}$$

  Such an estimated probability can be obtained in R as

```
InvLogit <- function(eta) 1/(1+exp(-eta))
as.numeric(InvLogit(c(1,0) %*% coef(fit1)))
```

$[1\ 0] \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} = \beta_0$

```
## [1] 0.6470185
```

```
predict(fit1, newdata=data.frame(dist = 0), type="response")
```

```
##         1
## 0.6470185
```

- Approximate confidence interval (of level 0.95%) for the parameter β_2

```
coef(fit1)[2] +c(-1,1) * summary(fit1)$coefficients[2,2] * qnorm(0.975)
```

```
## [1] -0.008128331 -0.004309307
```

- Interpreting the coefficients for the predictor `dist` on the log-odds scale: the coefficient for `dist` can be interpreted as a difference of -0.0062 in the logit probability of switching well when the distance is increased by one, that is

$$\text{logit}(P(Y_i = 1|dist_i = c + 1) = \beta_0 + (c + 1)\beta_1$$

$$\text{logit}(P(Y_i = 1|dist_i = c) = \beta_0 + c\beta_1$$

$$\text{logit}(P(Y_i = 1|dist_i = c + 1) - \text{logit}(P(Y_i = 1|dist_i = c) = \beta_1$$

- From the previous expression, we can use obtain the log-odds ratio

$$log\left(\frac{P(Y_i = 1|dist_i = c + 1)/P(Y_i = 0|dist_i = c + 1)}{P(Y_i = 1|dist_i = c)/P(Y_i = 0|dist_i = c)}\right) = \beta_1$$

and so

$$\frac{P(Y_i = 1|dist_i = c + 1)}{P(Y_i = 0|dist_i = c + 1)} = exp(\beta_1)\frac{P(Y_i = 1|dist_i = c)}{P(Y_i = 0|dist_i = c)}.$$

The ratio between the probability of success (in this case to switch well) and of failure (that is the odds), when $dist = c + 1$ is $exp(\beta_1) \approx 1$ times the odds when $dist = c$

```
exp(coef(fit1)[2])
```

```
##      dist
## 0.9938005
```

- Interpreting the coefficients for the predictor `dist` on the probability scale: we can evaluate the difference of the logistic regression function by adding 1 to the predictor `dist`. This difference corresponds to a difference on the probability scale but requires the choice of the specific value of the predictor, that is

$$P(Y_i = 1|dist_i = c+1)-P(Y_i = 1|dist_i = c) = \frac{1}{1 + exp(-(\beta_0 + (c + 1) \times \beta_1))} - \frac{1}{1 + exp(-(\beta_0 + c \times \beta_1))}$$

The mean of the predictor is a good starting point since it corresponds to the steepest point of the inverse logit function. Thus, adding 1 to`dist` (that is, adding 1 meters to the distance to the nearest safe well) corresponds to a negative difference in the probability of switching of about 0.15% on average. Of course, you can evaluate the probability between two different distances.

```
as.numeric(InvLogit(c(1, mean(data$dist) + 1) %*% coef(fit1))-
             InvLogit(c(1, mean(data$dist)) %*% coef(fit1)))
```

```
## [1] -0.001519722
```

```
as.numeric(InvLogit(c(1, 300) %*% coef(fit1))- InvLogit(c(1, 0) %*% coef(fit1)))
```

```
## [1] -0.4259907
```

Note that the coefficients here does not have a linear effect on the probability that the outcome is 1 because of the nonlinearity of the model. The curve of the logistic function requires us to choose where to evaluate changes, if we want to interpret on the probability scale. Below the difference on the probability of switching well due to an increase of one unit on the predictor from the 99th percentile

```
as.numeric(InvLogit(c(1, quantile(data$dist, 0.99) + 1) %*% coef(fit1))-
InvLogit(c(1, quantile(data$dist, 0.99)) %*% coef(fit1)))
```

```
## [1] -0.001465513
```

The effect of one unit increase of the variable `dist` on the inverse logit seems low, but this is misleading since distance is measured in meters, so this coefficient corresponds to the difference between, say, a house that is 48 meters away from the nearest safe well and a house that is 49 meters away. In order to improve interpretability of the results we rescale distance in 100-meter units. Thus, adding 100 meters to the distance to the nearest safe well corresponds to a negative difference in the probability of switching of about 15%.

```
data$dist100 <- data$dist/100
fit2 <- glm(switch ~ dist100 , data = data,
            family = binomial(link = "logit"))
summary(fit2)
```

```
##
## Call:
## glm(formula = switch ~ dist100, family = binomial(link = "logit"),
##     data = data)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.60596    0.06031  10.047  < 2e-16 ***
## dist100     -0.62188    0.09743  -6.383 1.74e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 4076.2  on 3018  degrees of freedom
## AIC: 4080.2
##
## Number of Fisher Scoring iterations: 4
```

```
as.numeric(InvLogit(c(1, mean(data$dist100) + 1) %*% coef(fit2))-
           InvLogit(c(1, mean(data$dist100)) %*% coef(fit2)))
```

```
## [1] -0.1542287
```

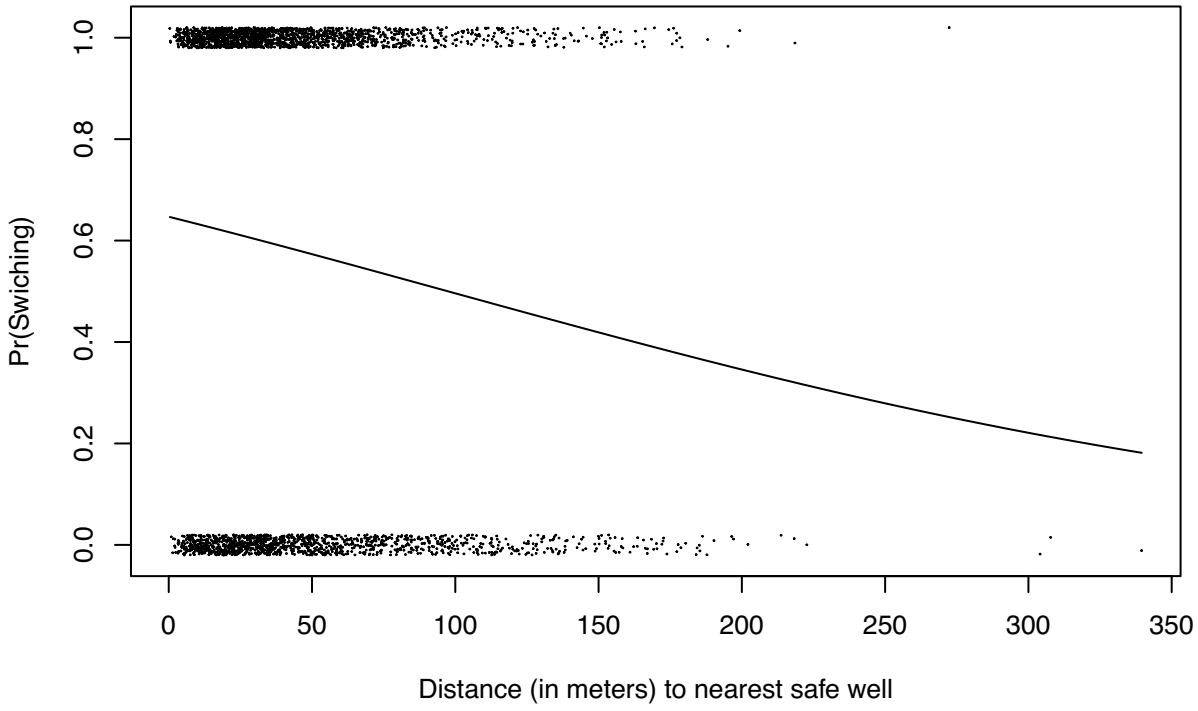
```
as.numeric(InvLogit(c(1, mean(data$dist) + 100) %*% coef(fit1))-
           InvLogit(c(1, mean(data$dist)) %*% coef(fit1)))
```

```
## [1] -0.1542287
```

The following plot provide information on the fitted model

```
switch.jitter <- jitter(data$switch, factor = 0.10)

par(mfrow = c(1, 1))
with(data, plot(dist, switch.jitter,
                xlab = "Distance (in meters) to nearest safe well",
                ylab = "Pr(Swiching)", type = "n"))
curve(InvLogit(coef(fit1)[1] + coef(fit1)[2] * x), lwd = 1, add = TRUE)
points(data$dist, switch.jitter, pch = 20, cex = .1)
```

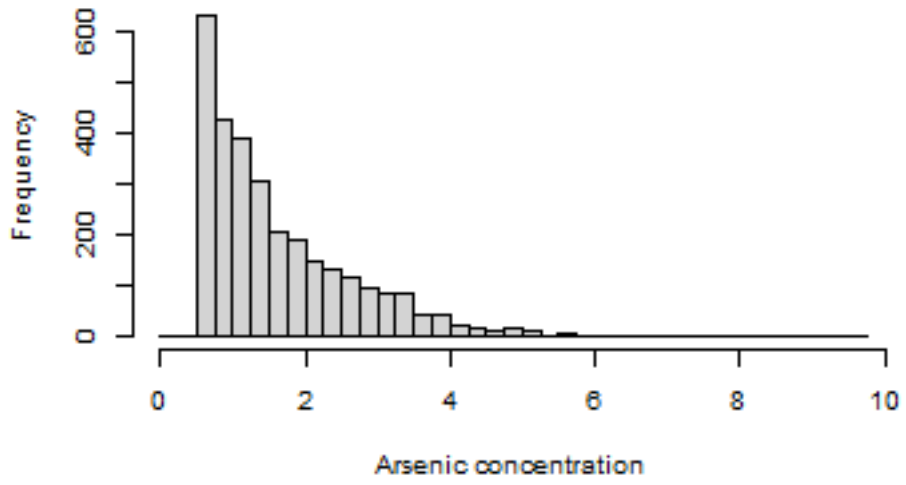


The probability of switching well is about 60% for those who live close to a safe well, declining to about 20% when the distance from a safe well increases up to 300 meters from their home. This seems reasonable: the probability of switching is higher for people who live closer to a safe well (Gelman-Hill, 2007).

Adding predictors

We consider adding the arsenic level in our model. We expect that the higher the arsenic concentration the higher the probability of switching well, that is, translated in model terms, a positive sign for the arsenic coefficient. We plot the histogram of arsenic concentration before fitting the model and analysing the results.

```
par(mfrow = c(1,1))
hist(data$arsenic, freq = TRUE, xlab = "Arsenic concentration",
     breaks = seq(0, 0.25 + max(data$arsenic), 0.25), main = "")
```




```
fit3 <- glm(switch ~ dist100 + arsenic, data = data, family = binomial("logit"))
summary(fit3)
```

```
##
## Call:
## glm(formula = switch ~ dist100 + arsenic, family = binomial("logit"),
##     data = data)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.002749   0.079448   0.035    0.972
## dist100     -0.896644   0.104347  -8.593   <2e-16 ***
## arsenic      0.460775   0.041385  11.134   <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: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3930.7  on 3017  degrees of freedom
## AIC: 3936.7
##
## Number of Fisher Scoring iterations: 4
```

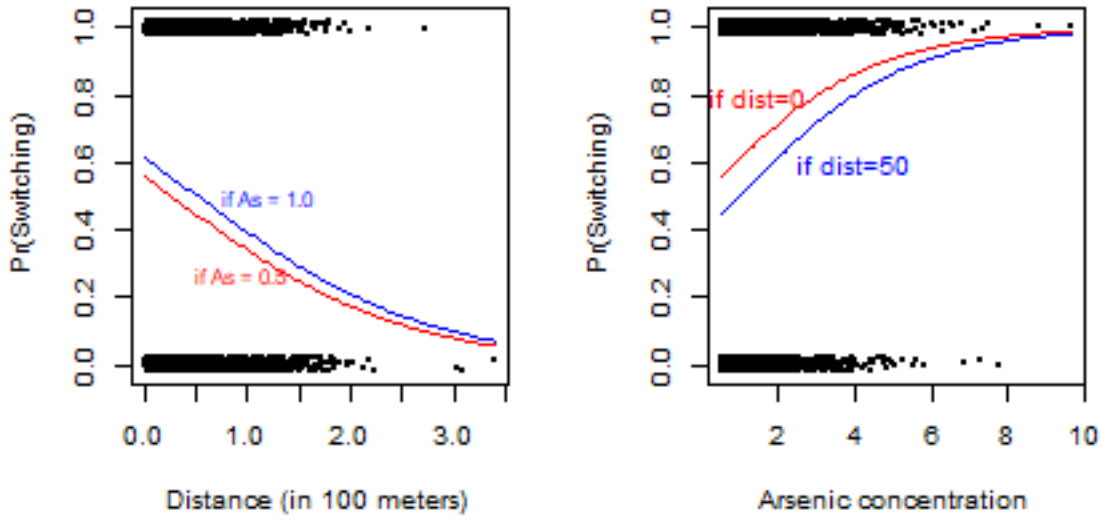
Numerical interpretation of one predictor effect here needs to be done choosing a value for the other predictor. As an example: given two wells with the same arsenic level (notice that it cannot be 0!), the *logit probability* of switching decreases of .9 every 100 meters in distance to the nearest safe well.

Equivalently: for two equally distant nearest safe wells, a difference of 1 in arsenic concentration corresponds to a 0.46 positive difference in the *logit probability* of switching.

The following plots show the fitted model with two predictors

```
par(mfrow = c(1, 2))
# Pr(Switching) vs distance (consider Arsenic = 0.5 and Arsenic = 1)
with(data, plot(dist100, switch.jitter,
                xlab = "Distance (in 100 meters)",
                ylab = "Pr(Switching)", type = "n"))
curve(InvLogit(coef(fit3)[1] + coef(fit3)[2] * x + coef(fit3)[3]), add = TRUE, col = "blue")
curve(InvLogit(coef(fit3)[1] + coef(fit3)[2] * x + coef(fit3)[3] * 0.5), add = TRUE, col = "red")
points(data$dist100, switch.jitter, pch = 20, cex = .1)
text (0.50, .27, "if As = 0.5", adj = 0, cex = .8, col = "red")
text (0.75, .50, "if As = 1.0", adj = 0, cex = .8, col = "blue")

# Pr(Switching) vs Arsenic (Consider distance = 0 and distance = 50)
with(data, plot(arsenic, switch.jitter,
                xlab = "Arsenic concentration",
                ylab = "Pr(Switching)", type = "n"))
curve(InvLogit(coef(fit3)[1] + coef(fit3)[3] * x), add = TRUE, col = "red")
text(1.5, 0.8, "if dist=0", col = "red")
curve(InvLogit(coef(fit3)[1] + coef(fit3)[2] * 0.5 + coef(fit3)[3] * x), add = TRUE, col = "blue")
text(4, 0.6, "if dist=50", col = "blue")
points(data$arsenic, switch.jitter, cex = 0.01, pch = 20)
```



In general terms, the fitted model can be summarized as: switching is easier if there is a nearby safe well, and if a household's existing well has a high arsenic level, there should be more motivation to switch.

Including interaction term

We now want to verify if there is a statistically significant effect of the interaction between the distance and the arsenic concentration on the probability of switching well.

```
fit4 <- glm(switch ~ dist100 + arsenic + dist100:arsenic,  #Equivalently (by using *)
            data = data, family = binomial(link = "logit"))
summary(fit4)
```

```
##
## Call:
## glm(formula = switch ~ dist100 + arsenic + dist100:arsenic, family = binomial(link = "logit"),
##     data = data)
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -0.14787    0.11754  -1.258  0.20838
## dist100         -0.57722    0.20918  -2.759  0.00579 **
## arsenic          0.55598    0.06932   8.021 1.05e-15 ***
## dist100:arsenic -0.17891    0.10233  -1.748  0.08040 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3927.6  on 3016  degrees of freedom
## AIC: 3935.6
##
## Number of Fisher Scoring iterations: 4
```

As said before, the interpretation of the coefficients when arsenic is equal to 0 does not make sense, since the minimum level reached is 0.5. We fix the predictors to their mean and so the probability of switching well needs to consider also the interaction term,

- The prob. of switching well for an average distance and an average arsenic concentration is about 0.59.

```
as.numeric(InvLogit(c(1,mean(data$dist100), mean(data$arsenic),
           mean(data$dist100)*mean(data$arsenic))%*%coef(fit4)))
```

```
## [1] 0.5868829
```

- At the mean level of arsenic in the data, each 100 meters of distance corresponds to an approximate 22% negative difference in probability of switching.

```
mean_d100 <- mean(data$dist100); mean_ars <- mean(data$arsenic)
as.numeric(InvLogit(c(1, mean_d100 + 1, mean_ars, (mean_d100+1) * mean_ars) %*%coef(fit4)) -
             InvLogit(c(1, mean_d100, mean_ars, mean_d100 * mean_ars) %*% coef(fit4)))
```

```
## [1] -0.2146287
```

- At the mean level of distance in the data, each additional unit of arsenic corresponds to an approximate 11% positive difference in probability of switching.

```
as.numeric(InvLogit(c(1, mean_d100, mean_ars + 1, mean_d100 * (mean_ars+1)) %*%coef(fit4)) -
             InvLogit(c(1, mean_d100, mean_ars, mean_d100 * mean_ars) %*% coef(fit4)))
```

```
## [1] 0.1074813
```

- The interaction term can be seen as the effect added to the coefficient for distance for each additional unit of arsenic: at the average level of arsenic, the coefficient for distance is -0.88. Viceversa, at the average level of distance, the coefficient for arsenic is 0.47. So, the importance of distance as a predictor increases for households with higher existing arsenic levels and the importance of arsenic decreases for households that are farther from existing safe wells.

```
coef(fit4)[2] + coef(fit4)[4]*mean(data$arsenic)
```

```
##    dist100
## -0.8736527
```

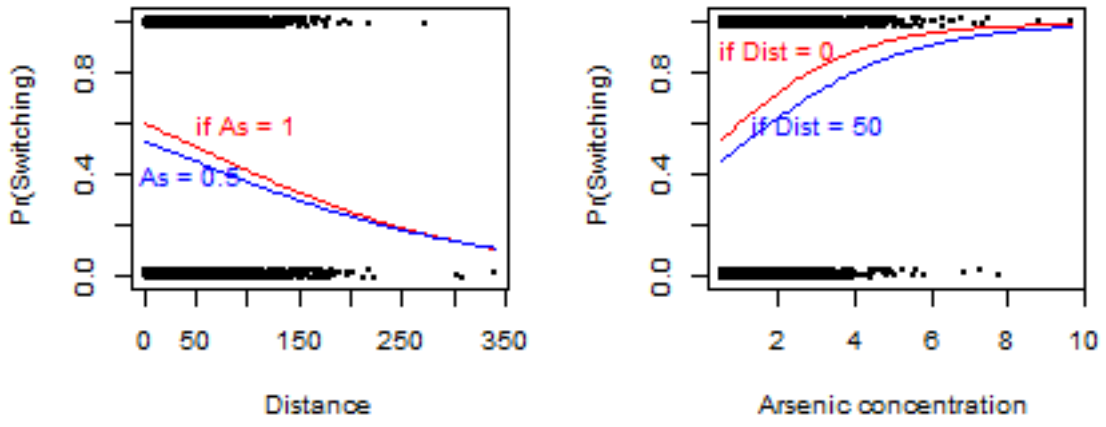
```
coef(fit4)[3] + coef(fit4)[4]*mean(data$dist100)
```

```
##   arsenic
## 0.4695081
```

The following plots show that the differences in switching associated with differences in arsenic level are larger if you are close to a safe well, but with a diminishing effect if you are far from any safe well. Comparing with the same plot without interaction, note that the two curves mainly differ in the area with few data points.

```
par(mfrow = c(1,2))
# Pr(Switching) vs Distance (Consider Arsenic=0.5 and Arsenic = 1)
with(data, plot(data$dist, switch.jitter, type = "n",
                xlab = "Distance", ylab = "Pr(Switching)"))
curve(InvLogit(cbind(1, x/100, 1, x/100) %*% coef(fit4)), add = TRUE, col = "red")
curve(InvLogit(cbind(1, x/100, 0.5, 0.5 * x/100) %*% coef(fit4)), add = TRUE, col = "blue")
text(100, 0.6,"if As = 1", col = "red"); text(35, 0.4,"if As = 0.5", col = "blue")
points(data$dist, switch.jitter, pch = 20, cex = 0.01)

# Pr(Switching) vs Arsenic (Consider Distance = 0 and Distance = 50)
with(data, plot(data$arsenic, switch.jitter, type = "n",
                xlab = "Arsenic concentration", ylab = "Pr(Switching)"))
curve(InvLogit(cbind(1, 0, x, 0) %*% coef(fit4)), add = TRUE, col = "red")
curve(InvLogit(cbind(1, 0.5, x, 0.5 * x) %*% coef(fit4)), add = TRUE, col = "blue")
text(2, 0.9, "if Dist = 0", col = "red"); text(3, 0.6, "if Dist = 50", col = "blue")
points(data$arsenic, switch.jitter, pch = 20, cex = 0.01)
```



Extra: Centering the variables

Interpreting the coefficients is easier when variables are centered (or eventually standardised). Coefficients are now interpretable as the effect of the predictor on the logit probability when the others are at their mean.

```
data$c.dist100 <- scale(data$dist100, scale = FALSE)
data$c.arsenic <- scale(data$arsenic, scale = FALSE)
# Refitting the fit3 and fit4 model with centered inputs
fit3b <- glm(switch ~ c.dist100 + c.arsenic, family = binomial(link = "logit"), data = data)
summary(fit3b)
```

```
##
## Call:
## glm(formula = switch ~ c.dist100 + c.arsenic, family = binomial(link = "logit"),
##     data = data)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.33286    0.03832   8.687   <2e-16 ***
## c.dist100   -0.89664    0.10435  -8.593   <2e-16 ***
## c.arsenic    0.46077    0.04138  11.134   <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: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3930.7  on 3017  degrees of freedom
## AIC: 3936.7
##
## Number of Fisher Scoring iterations: 4
```

β_1, β_2 are the same but β_0 changes

```
fit5 <- glm(switch ~ c.dist100 + c.arsenic + c.dist100:c.arsenic,
            family = binomial(link = "logit"), data = data); summary(fit5)
```

```
##
## Call:
## glm(formula = switch ~ c.dist100 + c.arsenic + c.dist100:c.arsenic,
##     family = binomial(link = "logit"), data = data)
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)           0.35109    0.03985   8.810   <2e-16 ***
## c.dist100            -0.87365    0.10480  -8.337   <2e-16 ***
## c.arsenic             0.46951    0.04207  11.159   <2e-16 ***
## c.dist100:c.arsenic -0.17891    0.10233  -1.748   0.0804 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3927.6  on 3016  degrees of freedom
## AIC: 3935.6
##
## Number of Fisher Scoring iterations: 4
```

Extra: Adding social predictors

We can add also the education variable in our model. However, in order to improve the interpretability of the results, we consider dividing the education variable by 4 and centering the predictor.

```
data$c.educ4 <- scale(data$educ/4, scale = FALSE)
fit6 <- glm(switch ~ c.dist100*c.arsenic + c.educ4, data = data,
            family = binomial(link = "logit"))
summary(fit6)
```

```
##
## Call:
## glm(formula = switch ~ c.dist100 * c.arsenic + c.educ4, family = binomial(link = "logit"),
##     data = data)
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)           0.35271    0.04001   8.816  < 2e-16 ***
## c.dist100            -0.87462    0.10510  -8.322  < 2e-16 ***
## c.arsenic             0.47663    0.04228  11.273  < 2e-16 ***
## c.educ4               0.16922    0.03833   4.415 1.01e-05 ***
## c.dist100:c.arsenic -0.16291    0.10235  -1.592    0.111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3907.9  on 3015  degrees of freedom
## AIC: 3917.9
##
## Number of Fisher Scoring iterations: 4
```

Moreover, we add the interaction between the new variable and the previous predictors `dist` and `arsenic`. The interpretation of the new interaction terms represents the effect of education on the effects of the other predictors. Positive changes in education reduce distance's negative association and increasing education increases arsenic's positive association.

```
fit7 <- glm(switch ~ c.dist100*c.arsenic + c.dist100*c.educ4 +
                     c.arsenic * c.educ4, data = data,
            family = binomial(link = "logit"))
summary(fit7)
```

```
##
## Call:
## glm(formula = switch ~ c.dist100 * c.arsenic + c.dist100 * c.educ4 +
##     c.arsenic * c.educ4, family = binomial(link = "logit"), data = data)
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)           0.35630    0.04028   8.844  < 2e-16 ***
## c.dist100            -0.90286    0.10731  -8.414  < 2e-16 ***
## c.arsenic             0.49498    0.04305  11.497  < 2e-16 ***
## c.educ4               0.18498    0.03919   4.720 2.36e-06 ***
## c.dist100:c.arsenic -0.11768    0.10353  -1.137  0.25569
## c.dist100:c.educ4    0.32269    0.10662   3.026  0.00247 **
```

```
## c.arsenic:c.educ4    0.07223    0.04387   1.647  0.09965 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3891.7  on 3013  degrees of freedom
## AIC: 3905.7
##
## Number of Fisher Scoring iterations: 4
```

Exercise Try to interpret the coefficients of the new model.

Model Comparison and Model Selection

Suppose that we partition our p-dimensional regression parameter vector as $\boldsymbol{\beta} = (\beta_1^\top, \beta_2^\top)^\top$ (with $dim(\beta_1) = p_0$ and $dim(\beta_2) = p - p_0$) and we want verify $H_0 : \boldsymbol{\beta}_1 = 0$ against $H_1 : \boldsymbol{\beta}_1 \neq 0$. So, the LRT based on the deviance difference is (asymptotically) distributed as $\chi^2_{p-p_0}$, and allows a model comparison.

In R we can do it:

```
#anova(fit1, test="Chisq")
# Same results - recall that fit2 consider dist/100
#anova(fit2, test="Chisq")
#anova(fit3, test="Chisq")
anova(fit4, test="Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: switch
##
## Terms added sequentially (first to last)
##
##
##                 Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                              3019     4118.1
## dist100          1   41.861      3018     4076.2 9.798e-11 ***
## arsenic          1  145.570      3017     3930.7 < 2.2e-16 ***
## dist100:arsenic  1    3.040      3016     3927.6   0.08124 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Same results - recall that fit5 consider the variable centered
#anova(fit5, test="Chisq")
```

The residuals deviance drops significantly adding the distance and the arsenic variables, while marginal benefits are obtained by considering the interaction effect.

The model selection can be carried out also by using the AIC. It judges a model by how close its fitted values tend to be to the true expected values, as summarized by a certain expected distance between the two. Because smaller AIC is better, the AIC penalizes a model for having many parameters.

```
AIC( glm(switch ~ 1, data = data,
           family = binomial(link = "logit")))
```

```
## [1] 4120.099
```

```
c(AIC(fit1), AIC(fit3), AIC(fit4))
```

```
## [1] 4080.238 3936.668 3935.628
```

The conclusions are similar to those reported above.

Exercise Consider also the model fit6 and fit7 and use the deviance for inferential comparisons of models and model selection.

Exercise Explore the **Anova**() function of the **car** package.

Binned Residuals

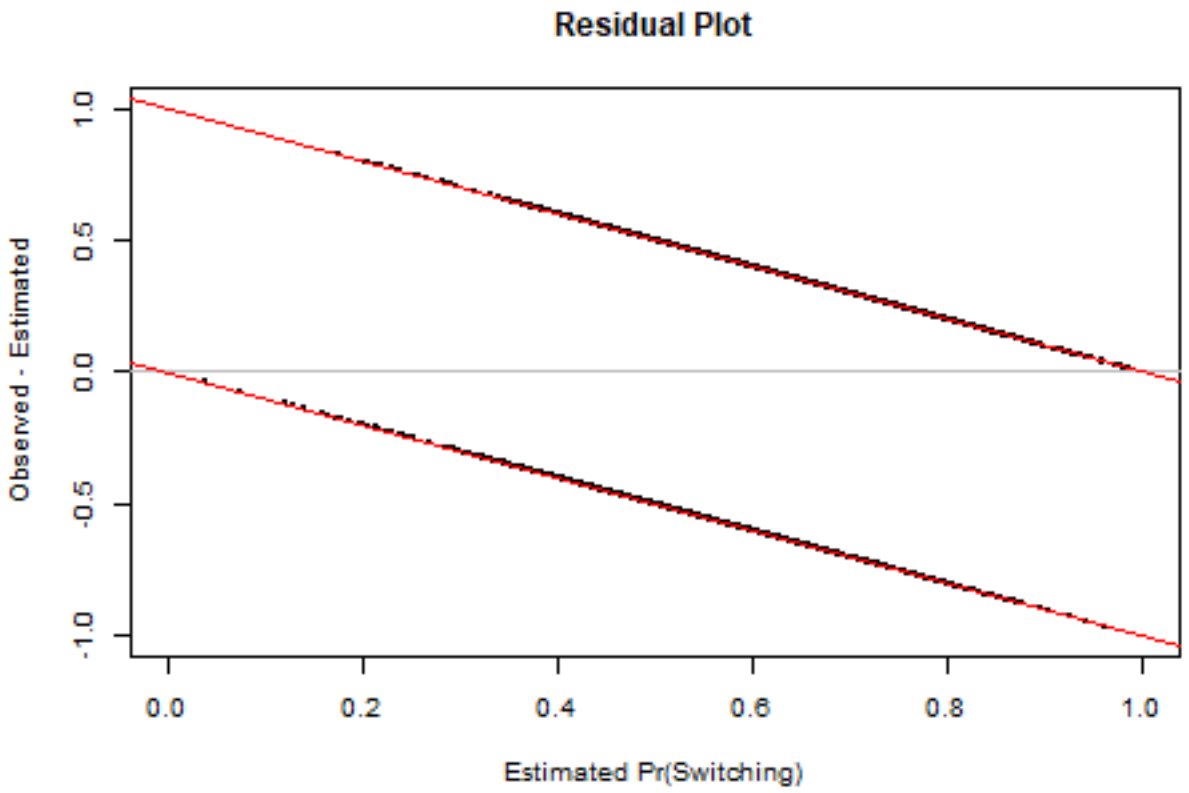
Outcome variable in the logistic regression can take only 0 and 1 values. The plot of the residuals defined as

$$residual_i = y_i - \hat{\pi}_i = y_i - \text{logit}^{-1}(\mathbf{x}_i^\top \hat{\beta})$$

versus the fitted values ($\hat{\pi} = \hat{E}(y_i|X_i)$) is not useful, that is considering $(\hat{\pi}_i, y_i - \hat{\pi}_i)$, since the points belong to two parallel lines with slopes -1.

```
pred7 <- predict(fit7, type="response")
res7 <- residuals(fit7, type="response")

plot(c(0,1),c(-1,1), xlab = "Estimated Pr(Switching)", type = "n",
     ylab = "Observed - Estimated", main = "Residual Plot")
abline(h = 0, col = "gray", lwd = 0.5)
points(pred7, res7, pch = 20, cex = 0.2)
abline(c(0,-1), col = "red", lwd = 0.25)
abline(c(1,-1), col = "red", lwd = 0.25)
```



A more readable plot can be made using the binned residuals defined as:

dividing the data into categories (bins) based on their fitted values, and then plotting the average residual versus the average fitted value for each bin (Gelman, Hill 2007).

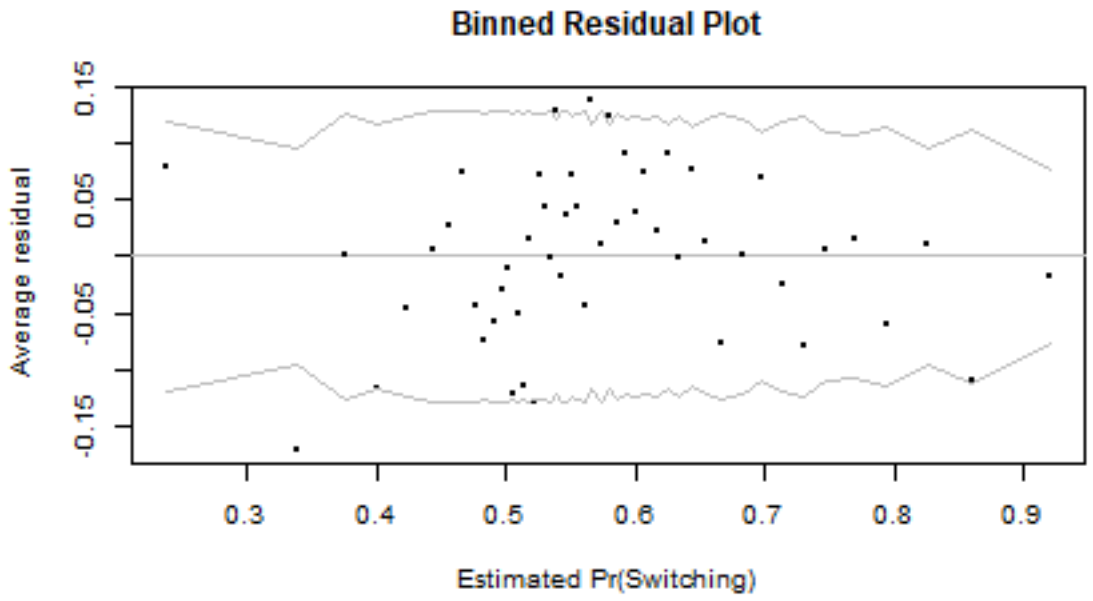
The number of bins have to be chosen such that each bin is computed on *enough* points such that the resulting plot is not too noisy (high number of bins) but can highlight patterns in the residuals (hided if the number of bins is too low).

Below, there is the function for obtaining the binned residuals (there are little changes w.r.t to the **binned.resids**() function of the **arm** package). The inputs are - `x`: represents the fitted values - `y`: represents the observed - fitted values - `nclass`: number of categories (bins)

```
binned.resids <- function(x, y, nclass = sqrt(length(x))){
 breaks.index <- floor(length(x) * (1 : (nclass))/nclass)
 breaks <- c(-Inf, sort(x)[breaks.index], Inf)
 output <- NULL
 xbreaks <- NULL
 x.binned <- as.numeric(cut(x, breaks))
 for(i in 1:nclass){
  items <- (1:length(x))[x.binned == i]
  x.range <- range(x[items])
  xbar <- mean(x[items])
  ybar <- mean(y[items])
  n <- length(items)
  sdev <- sd(y[items])
  output <- rbind(output, c(xbar, ybar, n, x.range, 2 * sdev/sqrt(n)))
 }
 colnames(output) <- c("xbar", "ybar", "n", "x.lo", "x.hi", "2se")
 return(list(binned = output, xbreaks = xbreaks))
}
```

The following plot shows the binned residuals vs the the estimated probability of switching. Grey lines depict ± 2 standard error (se) bounds, so we expect that 95% of the binned residuals falls between these two lines. The se is defined as $\sqrt{p(1-p)/n}$, where n is the number of data used to compute each average residual.

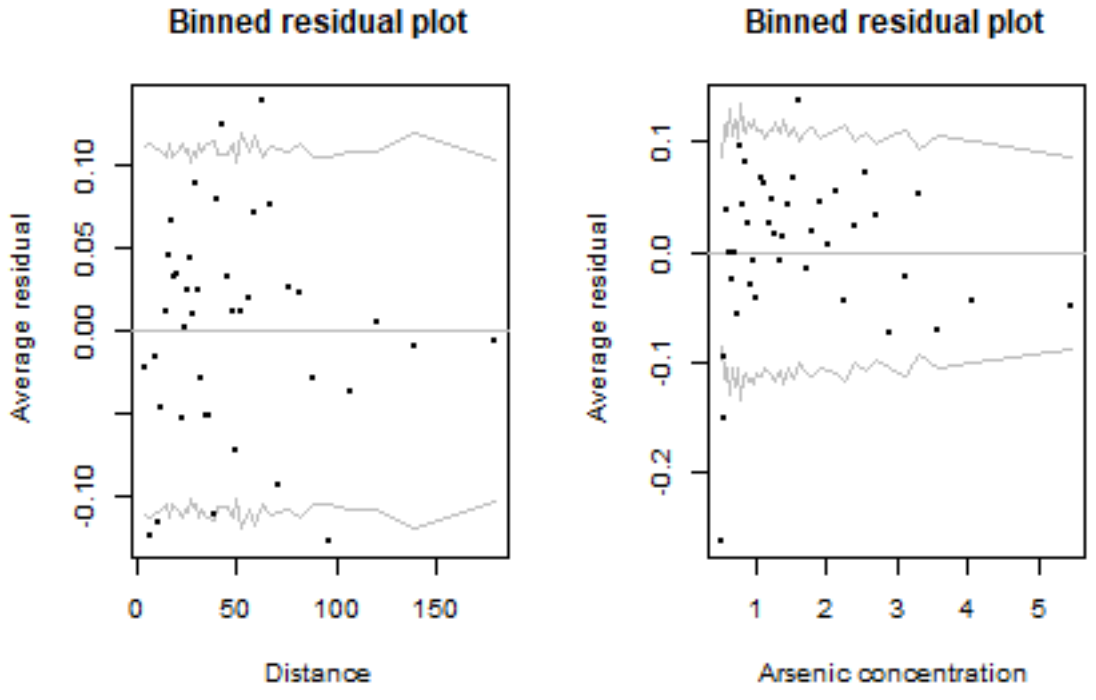
```
br7 <- binned.resids(pred7, res7, nclass = 50)$binned
plot(range(br7[,1]), range(br7[,2],br7[,6], -br7[,6]),  type = "n",
     xlab = "Estimated Pr(Switching)", ylab = "Average residual", main = "Binned Residual Plot")
abline(h = 0, col = "gray", lwd = 0.5); points(br7[,1], br7[,2], pch = 19, cex = 0.5)
lines(br7[,1], br7[,6], col = "gray", lwd = 0.5); lines(br7[,1], -br7[,6], col = "gray", lwd = 0.5)
```



A pattern can be identified looking at the plot of the binned residuals with respect to the arsenic level. Indeed, there are large negative binned residuals for some households with wells with small values of arsenic concentration. These points correspond to people that are less likely to switch well with respect to what predicted by the model (almost -20%). Moreover, the distribution of the residuals shows a slight pattern: the model seems to underestimate the probabilities of switching for medium arsenic level values while overestimates for high arsenic concentrations. So we plot the binned residuals vs the the predictors.

```
par(mfrow = c(1, 2))
br.dist <- binned.resids(data$dist, res7,
                         nclass = 40)$binned
plot(range(br.dist[,1]), range(br.dist[,2], br.dist[,6], -br.dist[,6]), type = "n",
     xlab = "Distance", ylab = "Average residual", main = "Binned residual plot")
abline(h = 0, col = "gray", lwd = 0.5)
lines(br.dist[,1], br.dist[,6], col = "gray", lwd = 0.5)
lines(br.dist[,1], -br.dist[,6], col = "gray", lwd = 0.5)
points(br.dist[,1], br.dist[,2], pch = 19, cex = 0.5)

br.arsenic <- binned.resids(data$arsenic, res7,
                         nclass = 40)$binned
plot(range(br.arsenic[,1]), range(br.arsenic[,2], br.arsenic[,6], -br.arsenic[,6]), type = "n",
     xlab = "Arsenic concentration", ylab = "Average residual", main = "Binned residual plot")
abline(h = 0, col = "gray", lwd = 0.5)
lines(br.arsenic[,1], br.arsenic[,6], col = "gray", lwd = 0.5)
lines(br.arsenic[,1], -br.arsenic[,6], col = "gray", lwd = 0.5)
points(br.arsenic[,1], br.arsenic[,2], pch = 19, cex = 0.5)
```



Exercise: Propose a possible solution to improve the model and fit your model. Compare the new binned residual plots with the previous ones.

Error rate

Error rate is defined as the proportion of cases for which the fitted probabilities are above 0.5 and the outcome value is 0 or the fitted probabilities are below 0.5 and the outcome value is 1, that is when the prediction-guessing is wrong:

$$er = mean((y = 0 \,\&\, \mathrm{E(y|X)} > .5) \,|\, (\mathrm{y} = 1 \,\&\, \mathrm{E(y|X)} < .5))$$

Just a point in the ROC curve
$\tau = 0.5$ ROC = ROC(τ)

It should be a value between 0 (perfect prediction-guessing) and 0.5 (random prediction-guessing).

Usually we expect the error rate to be lower than the error rate of the null model, that is the model including only the intercept. The estimated probability for the null model will be $p = \sum_i y_i/n$, that is the proportion of 1's in the data. The error rate for the null model is $min(p, 1-p)$, and it corresponds to assign 1 (or 0) to all the fitted values.

```
fit0 <- glm(switch ~ 1, family = binomial(link = "logit"), data = data)
m0 <- mean(data$switch)
err.rate0 <- min(m0, 1 - m0)
err.rate0
```

```
## [1] 0.4248344
```

```
# Alternatively
# 1 - InvLogit(coef(fit0))
```

So if our model improved the prediction-guessing of a simple logistic regression, we expect the error rate of our model to be lower than the error rate for the null model. Our last fitted model (model labelled as *fit7*), with an error rate of 0.38, correctly predicts 62% of the switching choices, only a 4% of improvement if compared to the simply guessing that all the households will switch.

```
err.rate7 <- mean((pred7 > 0.5 & data$switch == 0) |
                  ((pred7 < 0.5 & data$switch == 1 )))
err.rate7
```

Σ not in the diagonal

Similar to RSE w/ numerical variables

obtained in train set but we should test it in the test set

```
## [1] 0.3751656
```

Consider the error rate as the equivalent of the R^2 for the linear model, it can be a useful measure of the model fit but can't substitute a deeper evaluation of the model looking at coefficients significance and at the diagnostic plot. Analysing error rate means to extract information from the confusion matrix. Note that we can vary the threshold 0.5, and better assessment can be done by analysing the ROC curve.

```
conf_mat <- table(pred7 > 0.5, data$switch)
conf_mat
```

```
##
##            0    1
##   FALSE  438  288
##   TRUE   845 1449
```

```
sum(diag(conf_mat))/ nrow(data)
```

```
## [1] 0.6248344
```

```
1 - sum(diag(conf_mat))/ nrow(data)
```

```
## [1] 0.3751656
```

Exercise Compute the error rate of your proposed model and compare it with the previous one.

Example: Credit scoring

When issuing credit, banks check the "solvency" or "creditworthiness" of the client, i.e. the ability to pay back the credit. "Solvency" can be evaluated by a statistical model that tries to predict the probability that a credit is payed back by means of some economic or personal characteristics of the borrower. Binary regression models are suited for this situation.

We will consider a dataset of $n = 1000$ credits issued by a German bank. Every client is associated with a binary response defined as:

$$\begin{cases} y_i = 0 \text{ the client payed back his loan} \\ y_i = 1 \text{ the client did not pay back his loan} \end{cases}$$

Other characteristics were observed and could be used as covariates:

Variable name	Description
acc	no running account, bad running account, good running account
duration	duration of the credit in months
amount	credit amount in K-euros
moral	previous payment behaviour. 1 = good
intuse	intended use. 1 = private, 0 = business

Available information are in `credit1.txt` file. The data are now a data frame with 1000 rows and 7 columns. Let us give a look at the first 5 records:

```
Credit <- read.table("credit1.txt", header = TRUE)
str(Credit)
```

```
## 'data.frame':    1000 obs. of  6 variables:
##  $ y       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ acc     : chr  "no" "bad" "good" "good" ...
##  $ duration: int  24 12 18 12 24 24 36 24 21 10 ...
##  $ amount  : num  1.512 0.728 1.049 2.39 1.523 ...
##  $ moral   : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ intuse  : int  1 1 1 1 0 1 1 1 0 1 ...
```

```
Credit[1:5,]
```

```
##   y  acc duration    amount moral intuse
## 1 0   no       24 1.5118900     1      1
## 2 0  bad       12 0.7280797     1      1
## 3 0 good       18 1.0486600     1      1
## 4 0 good       12 2.3902900     1      1
## 5 0 good       24 1.5226270     1      0
```

First we may want to get an idea of the main variables. Note that 30% did not repaid the loan.

```
summary(Credit[-2])
```

```
##        y          duration        amount          moral           intuse
##  Min.   :0.0   Min.   : 4.0   Min.   :0.1278   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.0   1st Qu.:12.0   1st Qu.:0.6982   1st Qu.:1.000   1st Qu.:0.000
##  Median :0.0   Median :18.0   Median :1.1859   Median :1.000   Median :1.000
##  Mean   :0.3   Mean   :20.9   Mean   :1.6726   Mean   :0.911   Mean   :0.657
##  3rd Qu.:1.0   3rd Qu.:24.0   3rd Qu.:2.0310   3rd Qu.:1.000   3rd Qu.:1.000
##  Max.   :1.0   Max.   :72.0   Max.   :9.4200   Max.   :1.000   Max.   :1.000
```

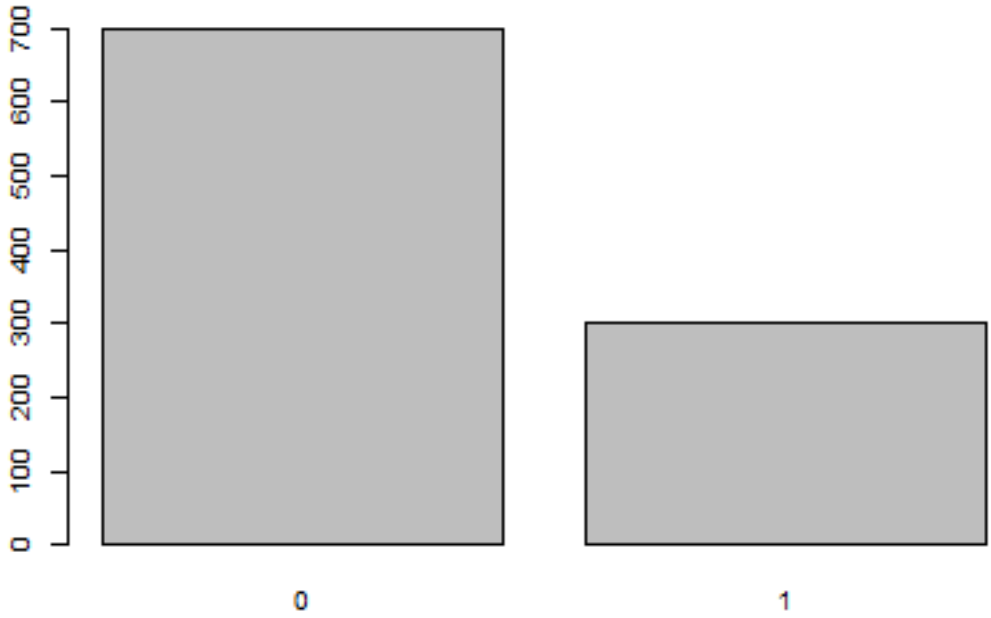
```r
table(Credit[2])
```

```
## acc
##  bad good   no
##  332  394  274
```

```r
table(Credit$y)
```

```
##
##   0   1
## 700 300
```

```r
barplot(table(Credit$y))
```



A logistic regression model

We can now try to estimate a first logistic regression model including all the avilable covariates. Note that we do not convert variables as `acc` is already a factor, while `moral` and `intuse` being dicothomous and so they can be handled as numeric without problems.

```r
mod1 <- glm(y ~ acc + duration + amount + moral + intuse,
            family = binomial(link = logit), data = Credit)
summary(mod1)
```

```
##
## Call:
## glm(formula = y ~ acc + duration + amount + moral + intuse, family = binomial(link = logit),
##     data = Credit)
```

```
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.284402   0.302579  -0.940 0.347255
## accgood     -1.337748   0.201127  -6.651 2.91e-11 ***
## accno        0.617659   0.174728   3.535 0.000408 ***
## duration     0.033233   0.007746   4.290 1.78e-05 ***
## amount       0.045875   0.064092   0.716 0.474134
## moral       -0.986066   0.250891  -3.930 8.49e-05 ***
## intuse      -0.425536   0.158272  -2.689 0.007174 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1221.7  on 999  degrees of freedom
## Residual deviance: 1029.8  on 993  degrees of freedom
## AIC: 1043.8
##
## Number of Fisher Scoring iterations: 4
```

Looking at the p-values we can see that all the variables are significantly different from zero except `amount` (actually to see if the `acc` effect is important, wed should consider the analysis of deviance).

The signs of the estimated coefficients are as expected. Remind that we are evaluating the effect of the covariates on the probability of not being creditworthy.

No running accounts seems to increase the probability of insolvency compared with those with a bad running account. While a good running account significantly affects the probability of being insolvent by decreasing it. We can evaluate the effect on the odds by computing

```
exp(mod1$coefficients[2])
```

```
##  accgood
## 0.262436
```

The odds of being insolvent, other things being equal, for those with a good account is around 0.25 times the odds of those with a bad account (or for those with a bad account is around 4 times the odds of those with a good account).

The coefficient associated with `duration` is significantly different from 0 and has a positive sign. This means that the longer is the term for repaying the debt and the higher is the probability of insolvency.

A more complex model

The model can be complicated by trying to see if there is a non linear effect of the two quantitative covariates. Now let us try the new model

```
mod2  <- glm(y ~ acc + duration + I(duration^2) + amount + I(amount^2) + moral + intuse,
             family = binomial(link = logit), data = Credit)
summary(mod2)
```

```
##
## Call:
## glm(formula = y ~ acc + duration + I(duration^2) + amount + I(amount^2) +
##     moral + intuse, family = binomial(link = logit), data = Credit)
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -0.4877826  0.3896495  -1.252 0.210625
## accgood       -1.3374022  0.2024364  -6.607 3.93e-11 ***
## accno          0.6178347  0.1761733   3.507 0.000453 ***
## duration       0.0921909  0.0252941   3.645 0.000268 ***
## I(duration^2) -0.0009094  0.0004133  -2.200 0.027781 *
## amount        -0.5165866  0.1926907  -2.681 0.007342 **
## I(amount^2)    0.0878646  0.0285982   3.072 0.002124 **
## moral         -0.9953315  0.2551618  -3.901 9.59e-05 ***
## intuse        -0.4039789  0.1601534  -2.522 0.011654 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1221.7  on 999  degrees of freedom
## Residual deviance: 1017.4  on 991  degrees of freedom
## AIC: 1035.4
##
## Number of Fisher Scoring iterations: 4
```

It seems we have improved our model (residual deviance is 1017.4, before was 1029.8; the AIC is 1035.4, before was 1043.8).

Predicting insolvency

A possible use of the model is to predict insolvency for a new client who wants to borrow a given amount from the bank. And the model could be used to decide to deny the loan if this probability is judged to be too high. In this case we assume that information on the covariates are available at the moment the bank has to decide.

Consider a client with the following characteristics: It has no running account, the term for repaying the debt is 36 months, the amount is 10000 euros, previous payment beahviour was bad and the money are intended to be use for business. How risky is to give a loan to him? The predicted probability of insolvency is:

```
client <- data.frame(acc = "no", duration = 36, amount = 10, moral = 0, intuse = 0)
predict(mod2, newdata = client, type = "response")
```

```
##         1
## 0.9972431
```

Our prediction shows that it could be very risky!

Extra example: Diabetes

We introduce here the dataset `PimaIndiansDiabetes2` with 768 observations on 9 variables. The goal is to predict the probability of being diabetes positive based on multiple clinical variables. The variables are:

Variable name	Description
pregnant	Number of times pregnant
glucose	Plasma glucose concentration (glucose tolerance test)
pressure	Diastolic blood pressure (mm Hg)
triceps	Triceps skin fold thickness (mm)
insulin	2-Hour serum insulin (mu U/ml)
mass	Body mass index (weight in kg/(height in m)^2)
pedigree	Diabetes pedigree function
age	Age (years)
diabetes	Class variable (test for diabetes)

Let's import and look at the data. We also remove some missing observations (here we are removing all the observations having at least a missing information in the record; however, note that we are almost halving the dataset and other possible solutions are suitable, such as removing variables with a large presence of missing values)

```
library(mlbench)
data("PimaIndiansDiabetes2", package = "mlbench")
apply(is.na(PimaIndiansDiabetes2), 2, sum)
```

```
## pregnant  glucose pressure  triceps  insulin     mass pedigree      age diabetes
##        0        5       35      227      374       11        0        0        0
```

```
PimaIndiansDiabetes2 <- na.omit(PimaIndiansDiabetes2)
```

We fit the model for the response variable `diabetes` using all the possible predictors:

```
mod_diabetes <- glm(diabetes ~ ., family = binomial, data = PimaIndiansDiabetes2)
summary(mod_diabetes)
```

```
##
## Call:
## glm(formula = diabetes ~ ., family = binomial, data = PimaIndiansDiabetes2)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.004e+01  1.218e+00  -8.246  < 2e-16 ***
## pregnant     8.216e-02  5.543e-02   1.482  0.13825
## glucose      3.827e-02  5.768e-03   6.635 3.24e-11 ***
## pressure    -1.420e-03  1.183e-02  -0.120  0.90446
## triceps      1.122e-02  1.708e-02   0.657  0.51128
## insulin     -8.253e-04  1.306e-03  -0.632  0.52757
## mass         7.054e-02  2.734e-02   2.580  0.00989 **
## pedigree     1.141e+00  4.274e-01   2.669  0.00760 **
## age          3.395e-02  1.838e-02   1.847  0.06474 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 498.10  on 391  degrees of freedom
## Residual deviance: 344.02  on 383  degrees of freedom
## AIC: 362.02
##
## Number of Fisher Scoring iterations: 5
```

According to the fit above, only three predictors are significantly different from zero: `glucose`, `mass` and `pedigree`.

Let's predict now the diabetes probability for a 60-years men, with 102 of glucose, 80 of diastolic pressure, body-mass index of 33 and 68 of triceps:

```
predict(mod_diabetes, newdata = data.frame(pregnant= 0,
          glucose = 102, pressure = 80, triceps = 68,
          insulin = 156, mass = 33, pedigree = 0, age = 60), type = "response")
```

```
##         1
## 0.2224012
```

The fitted model seems to be overparametrized: only three predictors out of eight are influential for the diabetes probability.

Exercise: Propose a better parsimonious model having similar performance.

Classification and prediction: train and test set

Now we will randomly split the data into training set (80% for building a predictive model) and test set (20% for evaluating the model). The goal is to measure the *predictive accuracy* of the model based on the test set. First of all we split the original dataset:

```
library(dplyr)
n <- length(PimaIndiansDiabetes2$diabetes)
set.seed(123)
train <- sample_n(PimaIndiansDiabetes2, 0.8 * n)
test <- setdiff(PimaIndiansDiabetes2, train)
```

Then, we fit the model on the train data:

```
full.model <- glm(diabetes ~ ., data = train, family = binomial)
```

Now we can compute the probabilities for the test set and defining a predictive rule: if $p > 0.5$ we predict a positive individual, negative otherwise.

```
probabilities <- predict(full.model, newdata = test, type = "response")
predicted.classes <- ifelse(probabilities > 0.5, "pos", "neg")
```

Finally we can compute the predictive accuracy on the test set (that is 1 - error rate):

```
observed.classes <- test$diabetes
mean(predicted.classes == observed.classes)
```

```
## [1] 0.8734177
```

Exercise The previous result could be compared with the predictive accuracy (on the test set) for different models. For instance, consider the parsimonious model obtained by the previous exercise and compare them.

Poisson Regression

Let's assume to have a response variable representing a count. A sensible choice is to model y_i, $i = 1, \ldots, n$, considering a Poisson distribution, which belongs to the exponential dispersion family and so a GLM can be fitted. The counts have a positive mean and it is sensible to consider a log-link function

$$g(\mu_i) = \log(\mu_i) = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j, \quad i = 1, \ldots, n$$

Then, the mean satisfies the

$$\mu_i = E[Y_i|\mathbf{x}_i] = \exp\left(\beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j\right)$$

The mean of Y at $x_{ij} = c+1$ is equal to the mean at $x_{ij} = c$ times $\exp(\beta_j)$, adjusted for the other explanatory variables, that is

$$E[Y_i|x_{ik} = c+1, \ldots] = \exp\left(\beta_0 + \sum_{j\neq k} x_{ij}\beta_j + (c+1)\beta_k\right)$$

$$E[Y_i|x_{ik} = c, \ldots] = \exp\left(\beta_0 + \sum_{j\neq k} x_{ij}\beta_j + c\beta_k\right)$$

$$\frac{E[Y_i|x_{ik} = c+1, \ldots]}{E[Y_i|x_{ik} = c, \ldots]} = \exp(\beta_k)$$

So the expected values of Y for a unit change in X e^{β_1} the expected values of X.

Example: Modelling Horseshoe Crab Satellite Counts

This example comes from Agresti and Kateri book (FSDS). The dataset refers to $n = 173$ female horseshoe crabs on a island between the coast of the Florida and the Gulf of Mexico.

During spawning other male crabs, called **satellites**, may cluster around the pair and also fertilize the eggs. The number of satellites **y**, for each female, is considered as outcome of interest. The available explanatory variable are

- the female crab's **color** (1 = medium light;2 = medium; 3 = medium dark; 4 = dark): a surrogate of the age of the crab, as older crab tends to have a darker color
- **spine** condition (1 = both good; 2 = one worn or broken; 3 = both worn or broken),
- **width** (in cm)
- **weight** (in kg)

We consider only the explanatory variables *weight* and *color* and and we fit the following GLM

```
load("Crabs.rda")
fit_poi1 <- glm(y ~ weight + factor(color), data = Crabs, family=poisson (link="log"))
summary(fit_poi1)
```

```
##
## Call:
## glm(formula = y ~ weight + factor(color), family = poisson(link = "log"),
##     data = Crabs)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -0.04978    0.23315  -0.214   0.8309
## weight          0.54618    0.06811   8.019 1.07e-15 ***
## factor(color)2 -0.20511    0.15371  -1.334   0.1821
## factor(color)3 -0.44980    0.17574  -2.560   0.0105 *
## factor(color)4 -0.45205    0.20844  -2.169   0.0301 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 632.79  on 172  degrees of freedom
## Residual deviance: 551.80  on 168  degrees of freedom
## AIC: 917.1
##
## Number of Fisher Scoring iterations: 6
```

One variable. To decide what to do we must analyze the deviance.

Comments:

- The estimated weight effect $\hat{\beta}_1 \approx 0.55$ means that the expected number of satellites increases as weight increases, adjusted for colors; that is, for a fixed level of color, the estimated expected number of satellites for a female crabs with $c+1$ kg is equal to the estimated expected number of satellites for a female crabs with c kg times $\exp(0.55) = 1.73$ and the Wald 95% confidence interval is $exp(0.546 \pm 1.96 \times 0.068)$, that is $(1.51, 1.97)$

$$\hat{E}[Y_i|weight_i = c+1, color_i = k] = \exp(0.55)\, \hat{E}[Y_i|weight_i = c, color_i = k]$$

```
exp(coef(fit_poi1)[2] + c(-1,1) * summary(fit_poi1)$coefficients[2, 2] * qnorm(0.975) )
```

```
## [1] 1.510864 1.973242
```

- Since the color variable is considered as categorical, the interpetation must be done w.r.t. to the reference category, here $color = 1$ (medium light). Thus, the estimates for the color categories suggest that the expected number of satellites decreases as color gets darker, adjusting for weights. For instance, given a value of weights the expected number of satellites for crabs of color 2 is $exp(-0.205) = 0.81$ times the expected number of satellites for crabs of color 1, that is

$$\hat{E}[Y_i|weight_i = c, color_i = 2] = \exp(-0.205)\, \hat{E}[Y_i|weight_i = c, color_i = 1]$$

Confidence intervals using ML invariance property

By analising the LRT we note that all the p-values are small (both considering the test $H_0 : \beta_1 = 0$ and the test $H_0 : \beta_2 = \beta_3 = \beta_4 = 0$), so both the variabes improve the fitting

```
# likelihood-ratio tests
anova(fit_poi1, test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: y
##
## Terms added sequentially (first to last)
##
##
##                Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                            172     632.79
## weight          1   71.925       171     560.87  < 2e-16 ***
## factor(color)   3    9.061       168     551.80  0.02848 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

But what if we compare the following GLMs by means of the AIC: the model with only factor is responsible of an improvement w.r.t. to the model with only an intercept, but it is clear that considering only the color in the model is insufficient; while the model only the weight behaves pretty good

```
fit_poi0 <- glm(y ~ 1 , data = Crabs, family=poisson (link="log"))
fit_poi2 <- glm(y ~ weight , data = Crabs, family=poisson (link="log"))
fit_poi3 <- glm(y ~ factor(color) , data = Crabs, family=poisson (link="log"))


AIC_comp<-rbind(AIC(fit_poi0), AIC(fit_poi3), AIC(fit_poi2), AIC(fit_poi1))
row.names(AIC_comp) <- c("int", "col", "wei","col+wei")
AIC_comp
```

```
##             [,1]
## int     990.0893
## col     972.4368
## wei     920.1641
## col+wei 917.1026
```

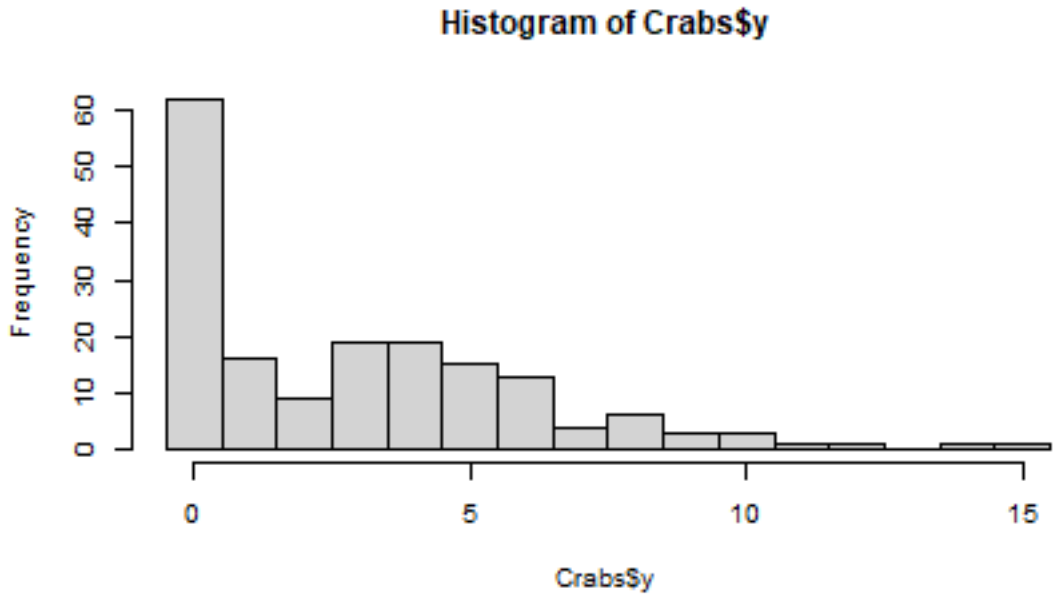
→ best model

Ok, we can use the other variables and considering the interactions (or other specifications), but maybe it is better to see the distribution of satellites, the sample mean and the sample variance. The empirical distribution shows the mode in 0, the variance (9.91) is larger than the mean (2.92), and this aspects represent strong evidence of over dispersion relative to the Poisson. Further, we analyse the Pearson's residuals of the fitted model.

```
table(Crabs$y)
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 14 15
## 62 16  9 19 19 15 13  4  6  3  3  1  1  1  1
```

```
hist(Crabs$y, breaks=c(0:16)-0.5)
```



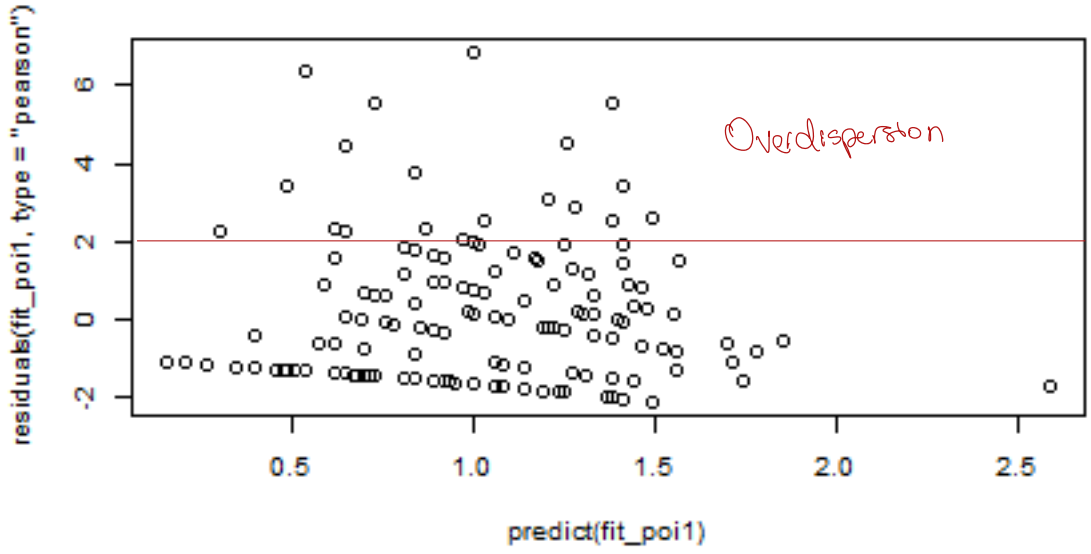

```
mean(Crabs$y)
```

```
## [1] 2.919075
```

```
var(Crabs$y)
```

```
## [1] 9.912018
```

```
par(mfrow = c(1,1))
plot(predict(fit_poi1), residuals(fit_poi1, type="pearson"))
```



Negative - Binomial Modelling

The Negative Binomial distribution is a Poisson- Gamma mixture that is a Poisson distribution where the parameter of the Poisson is a Gamma distribution. Leveraging on this relation we can obtain the negative binomial distribution, whose mean and variance are

$$E(Y) = \mu \qquad Var(Y) = \mu + \frac{\mu^2}{k}$$

(see Agresti and Kateri, pagg. 289-290 for further details). For the NB, $1/k$ is the dispersion parameter:

- as it increases, greater is the overdispersion relative to the Poisson
- as it approaches to 0, the variance decreases to the Poisson variance

We fit our Negative binomial regression model (log-link is the default) as follow:

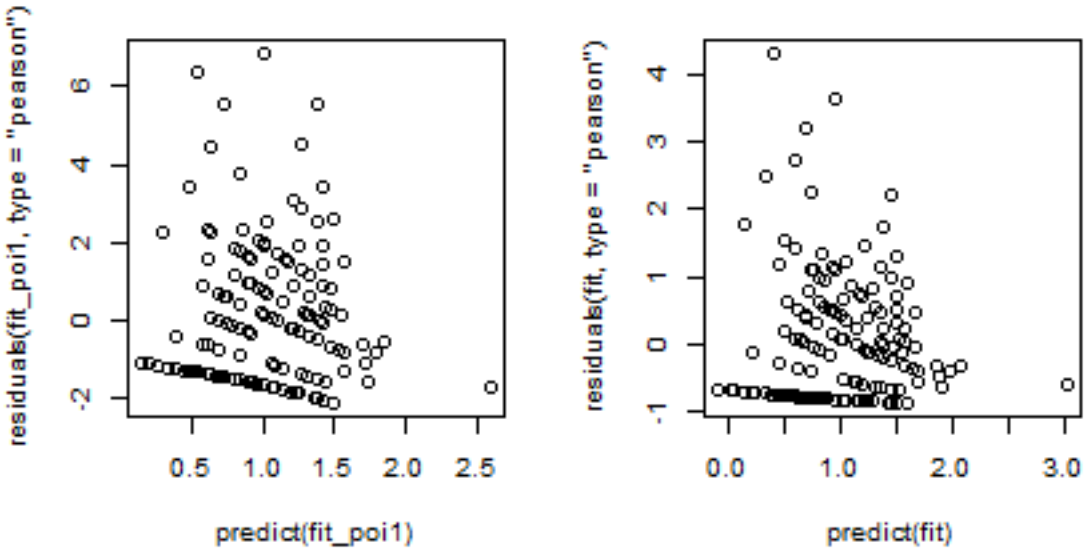
```
library(MASS)
fit <- glm.nb(y~ weight +factor(color),link=log,data=Crabs)
summary(fit)
```

```
##
## Call:
## glm.nb(formula = y ~ weight + factor(color), data = Crabs, link = log,
##     init.theta = 0.9596400657)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -0.4263     0.5382  -0.792    0.428
## weight           0.7121     0.1615   4.410 1.04e-05 ***
## factor(color)2  -0.2527     0.3486  -0.725    0.468
## factor(color)3  -0.5218     0.3799  -1.373    0.170
## factor(color)4  -0.4804     0.4282  -1.122    0.262
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(0.9596) family taken to be 1)
##
##     Null deviance: 220.01  on 172  degrees of freedom
## Residual deviance: 196.56  on 168  degrees of freedom
## AIC: 757.94
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  0.960
##           Std. Err.:  0.175
##
##  2 x log-likelihood:  -745.935
```

```r
library(car)
Anova(fit)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: y
##                LR Chisq Df Pr(>Chisq)
## weight          17.4542  1  2.943e-05 ***
## factor(color)    2.7368  3      0.434
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
par(mfrow = c(1,2))
plot(predict(fit_poi1), residuals(fit_poi1, type="pearson"))
plot(predict(fit), residuals(fit, type="pearson"))
```



Comments:

- tendency of the mean response to increase with weight, adjusting for color.
- no longer significance of the color effect
- The weight effect has standard error that increases from 0.068 in the Poisson model to 0.161 in the negative binomial model
- estimated dispersion parameter: 1.04 (In the summary output Theta is our k and so $1/Theta = 1.04$) , much grater than 0
- AIC: 757.94 (much lower than those obtained with Poisson GLMs)

Summarising, all these results favoring the Negative-Binomial regression model in the comparison with the Poisson GLMs (the Poisson distribution is not a good choice to analyse such data!).

Quasi - Poisson

By relaxing the parametric assumption of Y_i (substituing it with weaker second order assumptions), we can obtain the QuasiPoisson model. The quasi-likelihood approach allows to deal with overdispersion problems: it is possible to specify $var(Y_i)$ so that there is more variability with respect to the exponential family. In quasipoisson one should take into account that variance is modelled as $Var(Y_i) = \phi\mu_i$.

```
library(MASS)
fit_quasi <- glm(y~ weight + factor(color), family=quasipoisson, data=Crabs)
summary(fit_quasi)
```

```
##
## Call:
## glm(formula = y ~ weight + factor(color), family = quasipoisson,
##     data = Crabs)
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -0.04978    0.41600  -0.120    0.905
## weight          0.54618    0.12153   4.494  1.3e-05 ***
## factor(color)2 -0.20511    0.27427  -0.748    0.456
## factor(color)3 -0.44980    0.31356  -1.434    0.153
## factor(color)4 -0.45205    0.37191  -1.215    0.226
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 3.183714)
##
##     Null deviance: 632.79  on 172  degrees of freedom
## Residual deviance: 551.80  on 168  degrees of freedom
## AIC: NA : We don't have the family assumption to feed the likelihood function
##
## Number of Fisher Scoring iterations: 6
```

Comments:

- The coefficient estimates are the same of the Poisson model, since the estimating equations do not change.
- Instead, the standard errors of estimates will change since a value different from 1 is estimated for ϕ.

Exercise: What do you exoected analysing the residuals?

Example: Modelling claims frequency

The dataset *SingaporeAuto.txt* contains information on the number of accidents for a subset of data collected by the General Insurance Association of Singapore. Also some characteristics of the driver and the car are available. Only the data that refer to automobiles will be considered here.

The purpose of the analysis is to understand the impact of vehicle and driver characteristics on accident experience. These relationships represent the base for actuaries working in rate making, i.e., setting the price of the insurance coverages.

The variables and their description are the following

Variable name	Description
Female	Gender of Insured (=1 if female)
PC	=1 if private vehicle
Clm_Count	Number of claims during the year
Exp_weights	the fraction of the year the policy is in effect (an exposure variable)
NCD	No Claim discount: the higher the discount the better is the prior accident record
AgeCat	The age of the policy holder grouped into 7 categories (0,6). 0 = less than 21 and the other refer respectively to 22–25, 26–35, 36–45, 46–55, 56–65, over 66
VAgeCat	The age of the vehicle in 6 categories (0–6 indicat groups 0,1,2,3–5,6–10,11–15, 16 and older)

Thus, we read the data and we give a look at the first 5 records

```
accidents <- read.table("SingaporeAuto.txt",header=T)
accidents[1:5,]
```

```
##    Female PC Clm_Count Exp_weights NCD AgeCat VAgeCat
## 34      0  1         0  0.41889117  40      3       6
## 35      0  1         0  0.08487337  20      4       0
## 36      0  1         0  0.83778234  50      5       0
## 37      0  1         0  0.54757016  10      4       6
## 38      0  1         1  0.89527721  20      3       6
```

Note that the last 3 variables are actually categorical variables and can be convenient to redefine them. Note also that for automobile data not all the categories are observed.

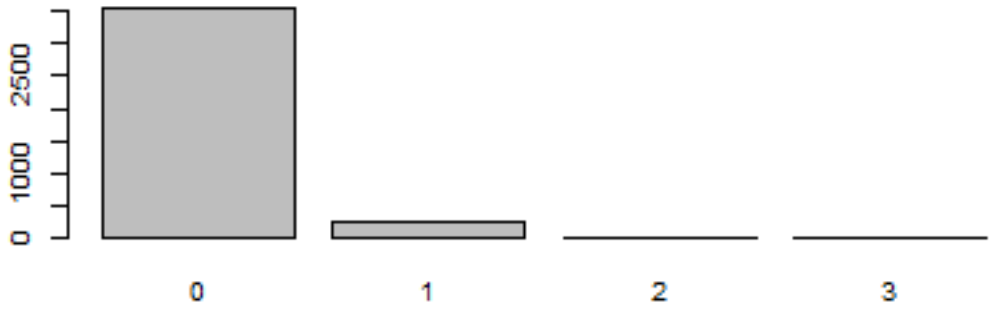
```
accidents$NCD <- as.factor(accidents$NCD)
accidents$AgeCat <- as.factor(accidents$AgeCat)
accidents$VAgeCat <- as.factor(accidents$VAgeCat)
```

The dependent variable is the count variable `Clm_Count`. Let first give a look at its distribution.

```
table(accidents$Clm_Count)
```

```
##
##    0    1    2    3
## 3555  271   15    1
```

```
barplot(table(accidents$Clm_Count))
```



We can now try to estimate a simple Poisson regression model

```
model1 <- glm(Clm_Count ~ Female + AgeCat + VAgeCat + NCD,
              family = poisson, data = accidents)
summary(model1)
```

```
##
## Call:
## glm(formula = Clm_Count ~ Female + AgeCat + VAgeCat + NCD, family = poisson,
##     data = accidents)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.46995    0.31001  -7.967 1.62e-15 ***
## Female      -0.15477    0.15528  -0.997  0.31890
## AgeCat3      0.26074    0.31539   0.827  0.40839
## AgeCat4      0.19969    0.32133   0.621  0.53430
## AgeCat5      0.03219    0.35342   0.091  0.92743
## AgeCat6      0.61281    0.39180   1.564  0.11780
## AgeCat7      0.78894    0.77462   1.018  0.30845
## VAgeCat1     0.27525    0.16972   1.622  0.10486
## VAgeCat2     0.46387    0.15848   2.927  0.00342 **
## VAgeCat6    -0.25227    0.71209  -0.354  0.72314
## NCD10       -0.21008    0.17788  -1.181  0.23761
## NCD20       -0.45123    0.21020  -2.147  0.03182 *
## NCD30       -0.42920    0.21532  -1.993  0.04623 *
## NCD40       -0.78419    0.26838  -2.922  0.00348 **
## NCD50       -0.64799    0.15494  -4.182 2.89e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1590.5  on 3841  degrees of freedom
## Residual deviance: 1551.8  on 3827  degrees of freedom
## AIC: 2166
##
```

```
## Number of Fisher Scoring iterations: 6
```

Taking into account exposure

We are not exactly modeling a count variable but a rate

The cars here considered have been insured only for a part of the year, and so we should take into account this because exposure is different for each car and cars less exposed have a lower number of accidents. We should rather model the rate, that is

$$\log(\mu_i/e_i) = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j \quad \text{equivalent to} \quad \log(\mu_i) = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j + \log(e_i)$$

where $\log(e_i)$ is called offset. In this specific case we want to model the rate $Clm\_Count/Exp\_weights$. But setting a log–linear model for this variable is equivalent at introducing into a Poisson regression model the logarithm of the variable (`Exp_weights`) as an *offset*. This can be done directly by writing:

```
model2 <- glm(Clm_Count ~ Female + AgeCat + VAgeCat + NCD, offset = log(Exp_weights),
              family = poisson, data = accidents)
summary(model2)
```

```
##
## Call:
## glm(formula = Clm_Count ~ Female + AgeCat + VAgeCat + NCD, family = poisson,
##     data = accidents, offset = log(Exp_weights))
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.65003    0.31027  -5.318 1.05e-07 ***
## Female      -0.17460    0.15524  -1.125  0.26072
## AgeCat3      0.16595    0.31540   0.526  0.59879
## AgeCat4      0.11903    0.32142   0.370  0.71115
## AgeCat5     -0.05699    0.35314  -0.161  0.87179
## AgeCat6      0.49974    0.39179   1.276  0.20212
## AgeCat7      0.69047    0.77848   0.887  0.37511
## VAgeCat1     0.19944    0.16977   1.175  0.24008
## VAgeCat2     0.40639    0.15863   2.562  0.01041 *
## VAgeCat6    -0.55839    0.71390  -0.782  0.43412
## NCD10       -0.24121    0.17828  -1.353  0.17605
## NCD20       -0.46202    0.20984  -2.202  0.02768 *
## NCD30       -0.46611    0.21597  -2.158  0.03091 *
## NCD40       -0.82628    0.26847  -3.078  0.00209 **
## NCD50       -0.72454    0.15521  -4.668 3.04e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1514.5  on 3841  degrees of freedom
## Residual deviance: 1472.6  on 3827  degrees of freedom
## AIC: 2086.8
##
## Number of Fisher Scoring iterations: 6
```

the model is better w/ the offset

Parameters do not change much, but if you compare the AIC values of the 2 models you will see that there is a reduction indicating that the second model should be preferred.

Exercise: Enhance the model, also accounting for possible overdispersion