Notebook 2: Supervised learning (regression)

Again, first define the path where the notebooks reside.

```
rm(list = ls(all = TRUE)) #clean up workspace
path <- "" #type the full path between the quotes and use
           #forward slashes ('/') to separate directories.
# Example:
path <- "C:/Users/koenn/OneDrive/School/DAV/Notebook 2"
options(digits = 3)
```

First, we load the data (a binary file).

```
load(file.path(path,"adsENG.RData"))
dat <- dat[1:2000,]
ls()
```

```
## [1] "dat"  "path"
```

The data consist of juvenile convicts in the Netherlands in 2006. Variables include sex (SEX), country of birth (COUNTRYBIRTH), age in years (AGE), age at first conviction in years (AGE1STCASE), crime type of the index case (CRIMETYPE), number of previous convictions (PREVCASES) and the time at risk in days, corrected for days spent imprisoned (EXPOSURE).

Have a look at the summary statistics

```
summary(dat)
```

```
##     SUMCASES         SEX                  COUNTRYBIRTH       AGE
##  Min.   : 0.00   Male  :1670   Netherlands      :1701   Min.   :12.0
##  1st Qu.: 0.00   Female: 330   Other non-Western: 105   1st Qu.:15.0
##  Median : 1.00                 Other Western    :  57   Median :16.0
##  Mean   : 1.28                 Dutch Antilles   :  46   Mean   :15.6
##  3rd Qu.: 2.00                 Morocco          :  35   3rd Qu.:17.0
##  Max.   :15.00                 (Other)          :  55   Max.   :21.0
##                                NA's             :   1
##    AGE1STCASE                        CRIMETYPE     PREVCASES
##  Min.   :11.0   property without violence:686   Min.   : 0.00
##  1st Qu.:14.0   public order             :590   1st Qu.: 0.00
##  Median :15.0   violence                 :337   Median : 0.00
##  Mean   :14.9   misc                     :136   Mean   : 0.72
##  3rd Qu.:16.0   property with violence   :106   3rd Qu.: 1.00
##  Max.   :21.0   (Other)                  :143   Max.   :15.00
##                 NA's                     :  2
##     EXPOSURE
##  Min.   :   1
##  1st Qu.:1461
##  Median :1461
##  Mean   :1443
##  3rd Qu.:1461
##  Max.   :1461
##
```

Remove cases with missing values.

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

Make new variable y containing number of cases per year. This will be our dependent variable. We will take the log, or else our model will be able to generate negative predictions. The + 1 is for preventing taking the log of zero.

```
dat$y <- log((dat$SUMCASES / dat$EXPOSURE * 365.25) + 1)
```

Split the data in training, validation and testing set. We will do this approximately in the ratio 60/20/20.

```
set.seed(13768) #for reproducability
randnum <- runif(nrow(dat)) #uniformly distributed random numbers in the range [0,1]
train <- dat[randnum<=0.6, ]
val <- dat[randnum >0.6 & randnum <= 0.8, ]
test <- dat[randnum > 0.8,]  #keep this data in a safe for when the modelling phase is over.
```

Fit the first simple linear regression model, dropping unwanted variables by using the minus sign.

```
mod1 <- lm(y ~ . -EXPOSURE -SUMCASES, data = train)
summary(mod1)
```

```
##
## Call:
## lm(formula = y ~ . - EXPOSURE - SUMCASES, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6575 -0.2041 -0.0648  0.1437  2.1170
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        0.63624    0.09136    6.96  5.4e-12
## SEXFemale                         -0.13436    0.02193   -6.13  1.2e-09
## COUNTRYBIRTHMorocco                0.17320    0.05978    2.90   0.0038
## COUNTRYBIRTHDutch Antilles        -0.04496    0.05404   -0.83   0.4056
## COUNTRYBIRTHSurinam                0.05878    0.07153    0.82   0.4114
## COUNTRYBIRTHTurkey                -0.03434    0.08184   -0.42   0.6749
## COUNTRYBIRTHOther Western         -0.11302    0.04787   -2.36   0.0184
## COUNTRYBIRTHOther non-Western     -0.04556    0.03567   -1.28   0.2018
## AGE                               -0.01341    0.01136   -1.18   0.2379
## AGE1STCASE                        -0.01270    0.01096   -1.16   0.2467
## CRIMETYPEsexual                    0.07318    0.05341    1.37   0.1708
## CRIMETYPEproperty with violence   -0.00234    0.04286   -0.05   0.9565
## CRIMETYPEproperty without violence  0.00513    0.02453    0.21   0.8343
## CRIMETYPEpublic order             -0.03488    0.02540   -1.37   0.1699
## CRIMETYPEdrugs                     0.00891    0.06009    0.15   0.8822
## CRIMETYPEtraffic                   0.01426    0.05168    0.28   0.7826
## CRIMETYPEmisc                     -0.10343    0.03826   -2.70   0.0070
## PREVCASES                          0.05943    0.00971    6.12  1.3e-09
##
## (Intercept)                       ***
## SEXFemale                         ***
## COUNTRYBIRTHMorocco               **
## COUNTRYBIRTHDutch Antilles
## COUNTRYBIRTHSurinam
## COUNTRYBIRTHTurkey
```

```
## COUNTRYBIRTHOther Western          *
## COUNTRYBIRTHOther non-Western
## AGE
## AGE1STCASE
## CRIMETYPEsexual
## CRIMETYPEproperty with violence
## CRIMETYPEproperty without violence
## CRIMETYPEpublic order
## CRIMETYPEdrugs
## CRIMETYPEtraffic
## CRIMETYPEmisc                      **
## PREVCASES                          ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.281 on 1198 degrees of freedom
## Multiple R-squared:  0.15,   Adjusted R-squared:  0.137
## F-statistic: 12.4 on 17 and 1198 DF,  p-value: <2e-16
```

Try to interpret the coefficients of the model. What does a unit of change in AGE accomplish on the scale of the outcome?

The in-sample (training) error MSE (still on transformed Y) is now calculated

```
mse <- function(obs, pred) mean((obs - pred)^2) #create function for calculation of mse
mAE <- function(obs, pred) mean(abs(obs - pred)) #mean absolute error
MAE <- function(obs, pred) median(abs(obs - pred)) #median absolute error
RSQ <- function(obs, pred) cor(obs, pred)^2 #Proportion of explained variance
yp.train <- predict(mod1)
mod1.mse.train <- mse(train$y, yp.train)
mod1.mAE.train <- mAE(train$y, yp.train)
mod1.MAE.train <- MAE(train$y, yp.train)
mod1.RSQ.train <- RSQ(train$y, yp.train)
```

Out-of-sample (validation) error MSE

```
yp.1.val <- predict(mod1, newdata = val)
mod1.mse.val <- mse(val$y, yp.1.val)
mod1.mAE.val <- mAE(val$y, yp.1.val)
mod1.MAE.val <- MAE(val$y, yp.1.val)
mod1.RSQ.val <- RSQ(val$y, yp.1.val)
cat(sprintf("mse training error model 1 is %.4f", mod1.mse.train),
    sprintf(", mse validation error is %.4f",mod1.mse.val), "\n", sep = "")
```

```
## mse training error model 1 is 0.0778, mse validation error is 0.0691
```

```
cat(sprintf("mAE training error model 1 is %.4f", mod1.mAE.train),
    sprintf(", mAE validation error is %.4f",mod1.mAE.val), "\n", sep = "")
```

```
## mAE training error model 1 is 0.2134, mAE validation error is 0.2102
```

```
cat(sprintf("MAE training error model 1 is %.4f", mod1.MAE.train),
    sprintf(", MAE validation error is %.4f",mod1.MAE.val), "\n", sep = "")
```

```
## MAE training error model 1 is 0.1846, MAE validation error is 0.1916
```

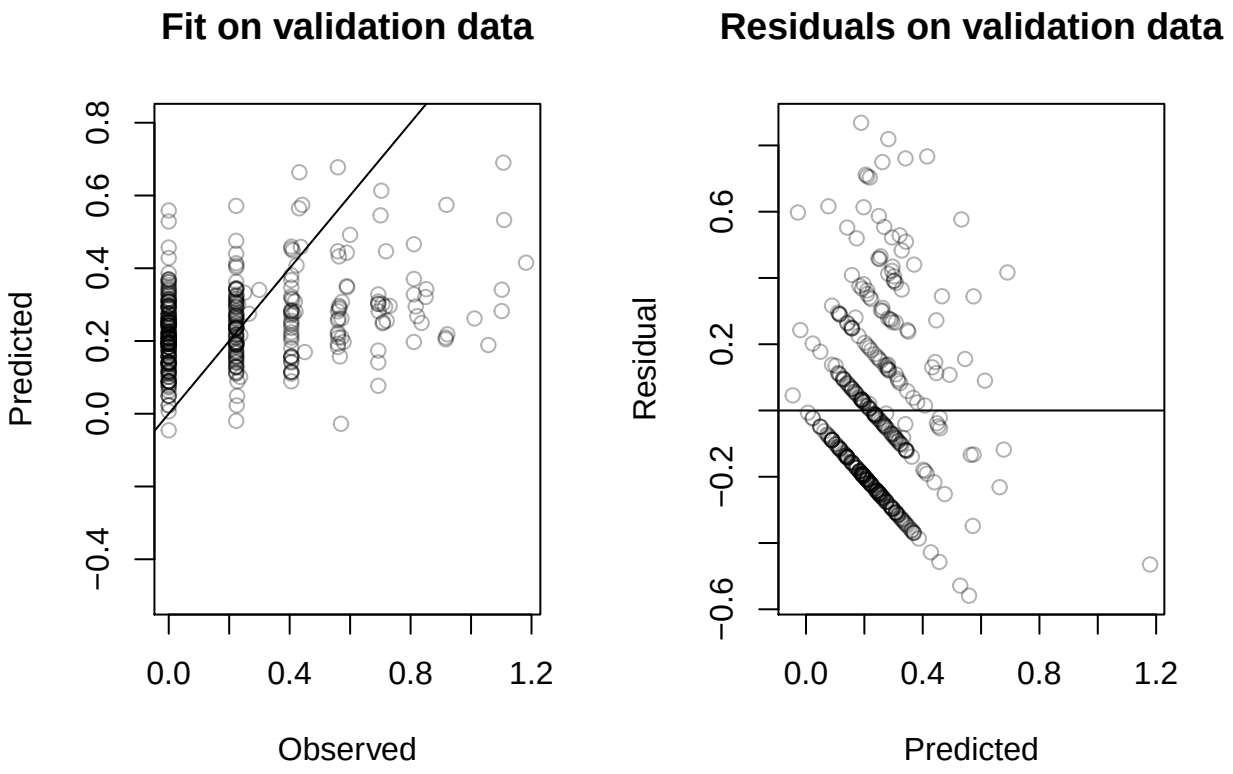
```
cat(sprintf("RSQ training error model 1 is %.4f", mod1.RSQ.train),
    sprintf(", RSQ validation error is %.4f",mod1.RSQ.val), "\n", sep = "")
```

```
## RSQ training error model 1 is 0.1495, RSQ validation error is 0.1158
```

The validation error and the training error are about the same.

Visualize observed versus predicted

```
trblack <-  rgb(0,0,0, alpha = 0.3) #create a slightly transparent black color.
                                    #Alpha stands for the opacity.
par(mfrow=c(1,2))
plot(val$y, yp.1.val, xlab = "Observed", ylab = "Predicted",
     main = "Fit on validation data", ylim = c(-0.5, 0.8), col = trblack)
abline(0,1)
plot(yp.1.val, val$y - yp.1.val, xlab = "Predicted", ylab = "Residual",
     main = "Residuals on validation data", col = trblack)#, ylim = c(-0.5, 0.8))
abline(h = 0)
```



Certainly not a good fit. The (raw) residual plot shows a dense region on the negative side. Until now, we have assumed that all variables are linearly related to the outcome.

From criminological research, it is known that the effect of number of previous convictions levels off after rising first (however this is for recidivism prevalence, not for frequency). Furthermore, a concave effect of age is known. If we transform these predictors, this might prevent the mismatch between the observed and predicted outcomes.

```
mod2 <- lm(y ~ SEX + COUNTRYBIRTH + AGE + I(AGE^2) +  AGE1STCASE +
             CRIMETYPE + log(PREVCASES + 1), data = train)
```

```
yp.train <- predict(mod2)
mod2.mse.train <- mse(train$y, yp.train)
```

```
yp.2.val <- predict(mod2, newdata = val)
mod2.mse.val <- mse(val$y, yp.2.val)
cat(sprintf("training error model 1 is %.4f", mod1.mse.train),
    sprintf(", validation error is %.4f",mod1.mse.val), "\n", sep = "") # \n means newline
```

```
## training error model 1 is 0.0778, validation error is 0.0691
```

```
cat(sprintf("training error model 2 is %.4f", mod2.mse.train),
    sprintf(", validation error is %.4f",mod2.mse.val), sep = "")
```

```
## training error model 2 is 0.0777, validation error is 0.0696
```

This only gives a slight improvement in the training data, but an exacerbation of error in the validation data.

Maybe fitting higher order polynomials will improve the fit.

```
mod3 <- lm(y ~ SEX + COUNTRYBIRTH + AGE*I(AGE ^ 2)*I(AGE ^ 3) +
             AGE1STCASE*I(AGE1STCASE ^ 2)*I(AGE1STCASE ^ 3) +
             CRIMETYPE + PREVCASES*I(PREVCASES^2)*I(PREVCASES^3) , data = train)
```

In-sample (training) error mse

```
yp.train <- predict(mod3)
mod3.mse.train <- mse(train$y, yp.train)
```

Out-of-sample (validation) error mse

```
yp.3.val <- suppressWarnings(predict(mod3, newdata = val))
mod3.mse.val <- mse(val$y, yp.3.val)
cat(sprintf("training error model 1 is %.4f", mod1.mse.train),
    sprintf(", validation error is %.4f",mod1.mse.val), "\n", sep = "")
```

```
## training error model 1 is 0.0778, validation error is 0.0691
```

```
cat(sprintf("training error model 2 is %.4f", mod2.mse.train),
    sprintf(", validation error is %.4f",mod2.mse.val), "\n", sep = "")
```

```
## training error model 2 is 0.0777, validation error is 0.0696
```

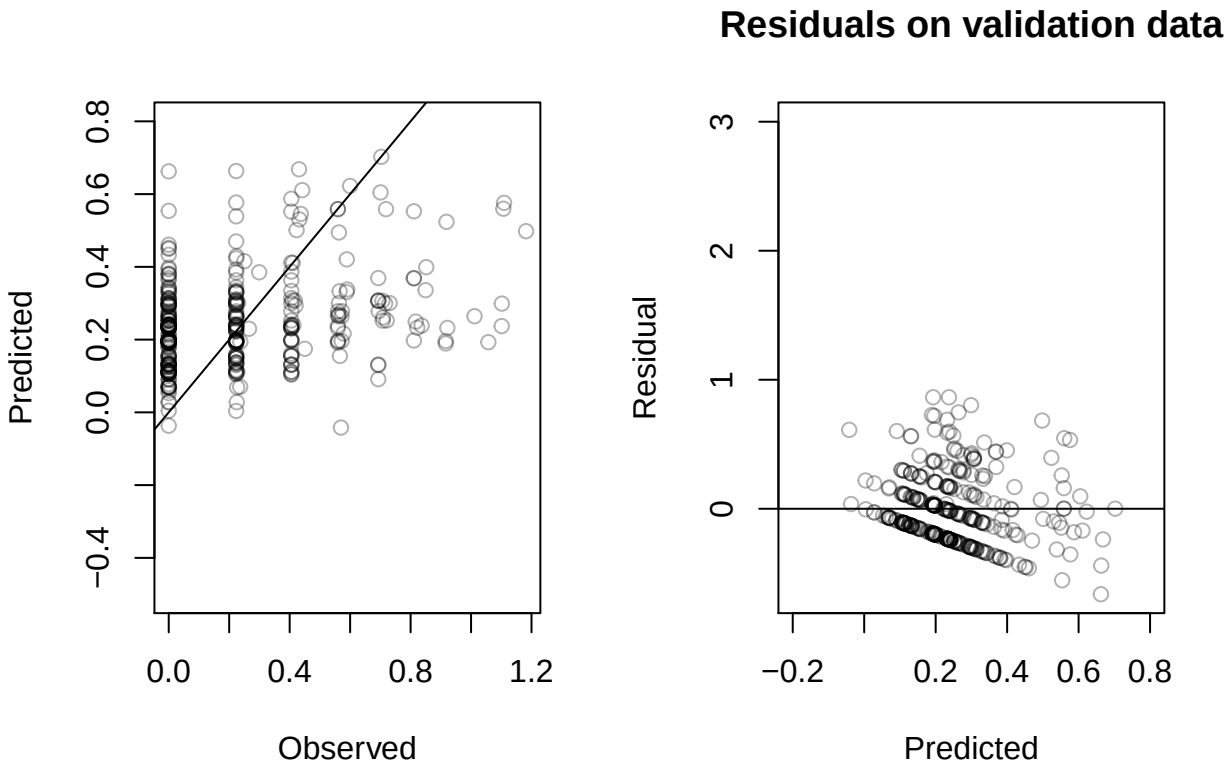
```
cat(sprintf("training error model 3 is %.4f", mod3.mse.train),
    sprintf(", validation error is %.4f",mod3.mse.val), sep = "")
```

```
## training error model 3 is 0.0763, validation error is 0.0958
```

we can now see the training error diminishes in the training set. However, the validation error rises significantly, indicating the presence of overfitting.

Again visualize observed versus predicted

```
par(mfrow = c(1,2))
plot(val$y, yp.3.val, xlab = "Observed", ylab = "Predicted", ylim = c(-0.5, 0.8), col=trblack)
abline(0,1)
plot(yp.3.val, val$y - yp.3.val, xlab = "Predicted", ylab = "Residual", main = "Residuals on validation
abline(h = 0)
```



Overfit can be prevented by regularization. In regularization, the minimization of the prediction error is combined with a simultaneous minimization of the size of the coefficients. Standardize (center and divide by the standard deviation) the data first to equate the influence of the penalty on the different coefficients.

First, the $L1$-penalty

```
msevec <- rep(0,5)
lambda <- c(0.001, 0.01, 0.1, 1, 10)  #lambda vector we want to try out

#the model formula
form <- y ~ SEX + COUNTRYBIRTH + AGE*I(AGE ^ 2)*I(AGE ^ 3) +
             AGE1STCASE*I(AGE1STCASE ^ 2)*I(AGE1STCASE ^ 3) +
             CRIMETYPE + PREVCASES*I(PREVCASES^2)*I(PREVCASES^3)
#library glmnet requires model formulation in the form of vectors/matrices
x.train <- model.matrix(form, data = train) #creates model matrix from formula object
x.val <- model.matrix(form, data = val)

library(glmnet)
library(dplyr)
for (i in 1:5){
paste("fitting with lambda = ", lambda[i])
# in package glmnet, $L1$- and $L2$-penalties can be mixed. The paramater alpha determines the mix.
#  Alpha = 1 is all L1, alpha = 0 is all L2.
mod <- glmnet(x = x.train, y = train$y, lambda = 1, alpha = 1, family = "gaussian",
              standardize = TRUE)
pred <- predict(mod, newx = x.val) #defaults to type = "response"
msevec[i] <- mse(val$y, pred)
```

```
}
cat("Best lambda is ", lambda[which.min(msevec)], "with mse = ", min(msevec), "\n")
```

```
## Best lambda is  0.001 with mse =  0.0769
```

```
#refit model
mod4 <- glmnet(x.train, train$y, lambda = lambda[which.min(msevec)], alpha = 1)
yp.4.train <- predict(mod4, newx = x.train, type = "response")
yp.4.val <- predict(mod4, newx = x.val, type = "response")
mod4.mse.train <- mse(train$y, yp.4.train)
mod4.mse.val <- mse(val$y, yp.4.val)
cat(sprintf("training error model 1 is %.4f", mod1.mse.train),
    sprintf(", validation error is %.4f",mod1.mse.val), "\n", sep = "")
```

```
## training error model 1 is 0.0778, validation error is 0.0691
```

```
cat(sprintf("training error model 3 is %.4f", mod3.mse.train),
    sprintf(", validation error is %.4f",mod3.mse.val), "\n", sep = "")
```

```
## training error model 3 is 0.0763, validation error is 0.0958
```

```
cat(sprintf("training error model 4 is %.4f", mod4.mse.train),
    sprintf(", validation error is %.4f",mod4.mse.val), "\n", sep = "")
```

```
## training error model 4 is 0.0771, validation error is 0.0691
```

```
coef(mod4) %>% zapsmall
```

```
## 37 x 1 sparse Matrix of class "dgCMatrix"
##                                            s0
## (Intercept)                             0.672
## (Intercept)                             .
## SEXFemale                              -0.130
## COUNTRYBIRTHMorocco                     0.163
## COUNTRYBIRTHDutch Antilles             -0.040
## COUNTRYBIRTHSurinam                     0.053
## COUNTRYBIRTHTurkey                     -0.028
## COUNTRYBIRTHOther Western              -0.105
## COUNTRYBIRTHOther non-Western          -0.042
## AGE                                    -0.013
## I(AGE^2)                                .
## I(AGE^3)                                .
## AGE1STCASE                             -0.017
## I(AGE1STCASE^2)                         .
## I(AGE1STCASE^3)                         .
## CRIMETYPEsexual                         0.070
## CRIMETYPEproperty with violence         .
## CRIMETYPEproperty without violence      0.003
## CRIMETYPEpublic order                  -0.035
## CRIMETYPEdrugs                          0.001
## CRIMETYPEtraffic                        0.005
## CRIMETYPEmisc                          -0.099
## PREVCASES                               0.082
## I(PREVCASES^2)                          .
## I(PREVCASES^3)                         -0.001
## AGE:I(AGE^2)                            .
## AGE:I(AGE^3)                            .
```

```
## I(AGE^2):I(AGE^3)                              .
## AGE1STCASE:I(AGE1STCASE^2)                     .
## AGE1STCASE:I(AGE1STCASE^3)                     .
## I(AGE1STCASE^2):I(AGE1STCASE^3)                .
## PREVCASES:I(PREVCASES^2)                       0.000
## PREVCASES:I(PREVCASES^3)                       .
## I(PREVCASES^2):I(PREVCASES^3)                  .
## AGE:I(AGE^2):I(AGE^3)                          0.000
## AGE1STCASE:I(AGE1STCASE^2):I(AGE1STCASE^3)     0.000
## PREVCASES:I(PREVCASES^2):I(PREVCASES^3)        0.000
```

We can see that $L1$-penalization remedies the overfit found in the overparameterized model (model 3). The MSE of the $L1$-penalized model is still however equal to that of the original model (model 1) on the validation set.

Now, we try the $L2$-penalty

```
library(magrittr) #for %>% operator
msevec <- rep(0,5)
lambda <- c(0.001, 0.01, 0.1, 1, 10)  #lambda vector we want to try out

library(glmnet)

for (i in 1:5){
paste("fitting with lambda = ", lambda[i])
# in package glmnet, $L1$- and $L2$-penalties can be mixed.
#  The parameter alpha determines the amount of mix between 0 and 1.
#  Alpha = 1 is all L1, alpha = 0 is all L2.
mod <- glmnet(x = x.train, y = train$y, lambda = 1, alpha = 0, family = "gaussian", standardize = TRUE)
pred <- predict(mod, newx = x.val) #defaults to type = "response"
msevec[i] <- mse(val$y, pred)
}
cat("Best lambda is ", lambda[which.min(msevec)], "with mse = ", min(msevec))
```

```
## Best lambda is  0.001 with mse =  0.0703
```

```
#refit model
mod5 <- glmnet(x.train, train$y, lambda = lambda[which.min(msevec)], alpha = 0)

#show nonzero coefficients
coef(mod5) %>% .[,1] %>% .[.!=0] %>% cbind %>% zapsmall
```

```
##                                              .
## (Intercept)                              0.789
## SEXFemale                               -0.131
## COUNTRYBIRTHMorocco                      0.167
## COUNTRYBIRTHDutch Antilles              -0.050
## COUNTRYBIRTHSurinam                      0.061
## COUNTRYBIRTHTurkey                      -0.042
## COUNTRYBIRTHOther Western               -0.112
## COUNTRYBIRTHOther non-Western           -0.047
## AGE                                     -0.006
## I(AGE^2)                                 0.000
## I(AGE^3)                                 0.000
## AGE1STCASE                              -0.029
## I(AGE1STCASE^2)                         -0.001
```

```
## I(AGE1STCASE^3)                                0.000
## CRIMETYPEsexual                                0.078
## CRIMETYPEproperty with violence                0.002
## CRIMETYPEproperty without violence             0.006
## CRIMETYPEpublic order                         -0.036
## CRIMETYPEdrugs                                 0.010
## CRIMETYPEtraffic                               0.012
## CRIMETYPEmisc                                 -0.100
## PREVCASES                                      0.075
## I(PREVCASES^2)                                 0.010
## I(PREVCASES^3)                                -0.001
## AGE:I(AGE^2)                                   0.000
## AGE:I(AGE^3)                                   0.000
## I(AGE^2):I(AGE^3)                              0.000
## AGE1STCASE:I(AGE1STCASE^2)                     0.000
## AGE1STCASE:I(AGE1STCASE^3)                     0.000
## I(AGE1STCASE^2):I(AGE1STCASE^3)                0.000
## PREVCASES:I(PREVCASES^2)                      -0.001
## PREVCASES:I(PREVCASES^3)                       0.000
## I(PREVCASES^2):I(PREVCASES^3)                  0.000
## AGE:I(AGE^2):I(AGE^3)                          0.000
## AGE1STCASE:I(AGE1STCASE^2):I(AGE1STCASE^3)     0.000
## PREVCASES:I(PREVCASES^2):I(PREVCASES^3)        0.000
```

```
yp.5.train <- predict(mod5, newx = x.train, type = "response")
yp.5.val <- predict(mod5, newx = x.val, type = "response")
mod5.mse.train <- mse(train$y, yp.5.train)
mod5.mse.val <- mse(val$y, yp.5.val)
cat(sprintf("training error model 1 is %.4f", mod1.mse.train),
    sprintf(", validation error is %.4f",mod1.mse.val), "\n", sep = "")
```

```
## training error model 1 is 0.0778, validation error is 0.0691
```

```
cat(sprintf("training error model 3 is %.4f", mod3.mse.train),
    sprintf(", validation error is %.4f",mod3.mse.val), "\n", sep = "")
```

```
## training error model 3 is 0.0763, validation error is 0.0958
```

```
cat(sprintf("training error model 4 is %.4f", mod4.mse.train),
    sprintf(", validation error is %.4f",mod4.mse.val), "\n", sep = "")
```

```
## training error model 4 is 0.0771, validation error is 0.0691
```

```
cat(sprintf("training error model 5 is %.4f", mod5.mse.train),
    sprintf(", validation error is %.4f",mod5.mse.val), "\n", sep = "")
```

```
## training error model 5 is 0.0768, validation error is 0.0715
```

$L2$-penalization (model 5) works less well than $L1$-penalization on these data. The best model found thus far is however still model 1 (because it is the simplest). Up to this point, we have used the validation data set to determine which model predicted the most accurate. However, while trying out diverse types of modelling, we can run the danger of overfitting the validation data set.

This is where the testing data comes in. After all modelling is finished, the final verdict can be computed on the testing data set.

```
# Best model found thus far
yp.1.test <- predict(mod1, newdata = test)
```

```
mod1.mse.test <- mse(test$y, yp.1.test)
yp.2.test <- predict(mod2, newdata = test)
mod2.mse.test <- mse(test$y, yp.2.test)
yp.3.test <- predict(mod3, newdata = test)
mod3.mse.test <- mse(test$y, yp.3.test)
x.test <- model.matrix(form, data = test)
yp.4.test <- predict(mod4, newx = x.test)
mod4.mse.test <- mse(test$y, yp.4.test)
yp.5.test <- predict(mod5, newx = x.test)
mod5.mse.test <- mse(test$y, yp.5.test)
c(mod1.mse.test,mod2.mse.test,mod3.mse.test,mod4.mse.test,mod5.mse.test)
```

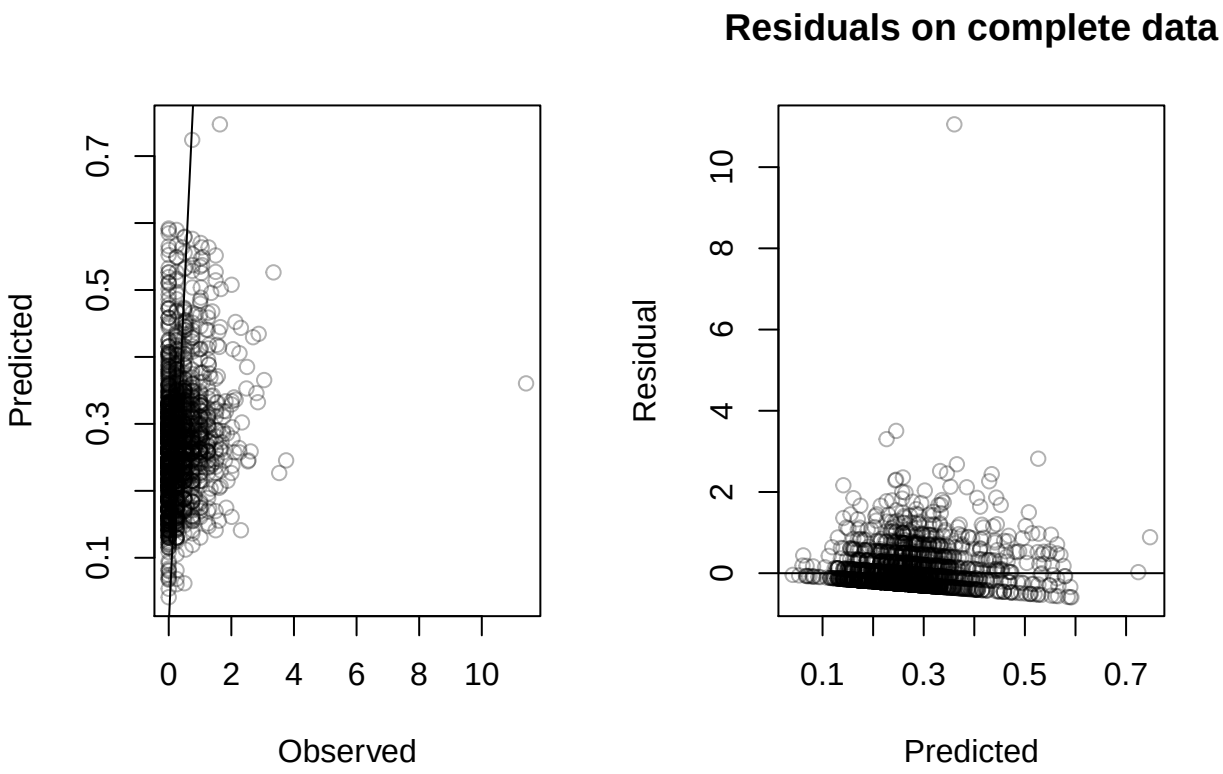
```
## [1] 0.0786 0.0791 0.0798 0.0784 0.0799
```

So, the test set points to model 4, the $L1$-penalized regression. After selecting this model, the final model will be fitted on the complete data set.

```
x <- model.matrix(form, data = rbind(train,test,val))
final <- glmnet(x = x, y = c(train$y,val$y,test$y), lambda = 0.001, alpha = 1, family = "gaussian", sta
yp <- predict(final, newx = x)
```

and now we plot the residuals on the original scale of the y-variable

```
transformback <- function(y) exp(y) - 1
par(mfrow = c(1,2))
plot(transformback(c(train$y,val$y,test$y)), transformback(yp),
     xlab = "Observed", ylab = "Predicted", col=trblack)
abline(0,1)
plot(transformback(yp), transformback(c(train$y,val$y,test$y)) - transformback(yp),
     xlab = "Predicted", ylab = "Residual", main = "Residuals on complete data", col = trblack)
abline(h = 0)
```



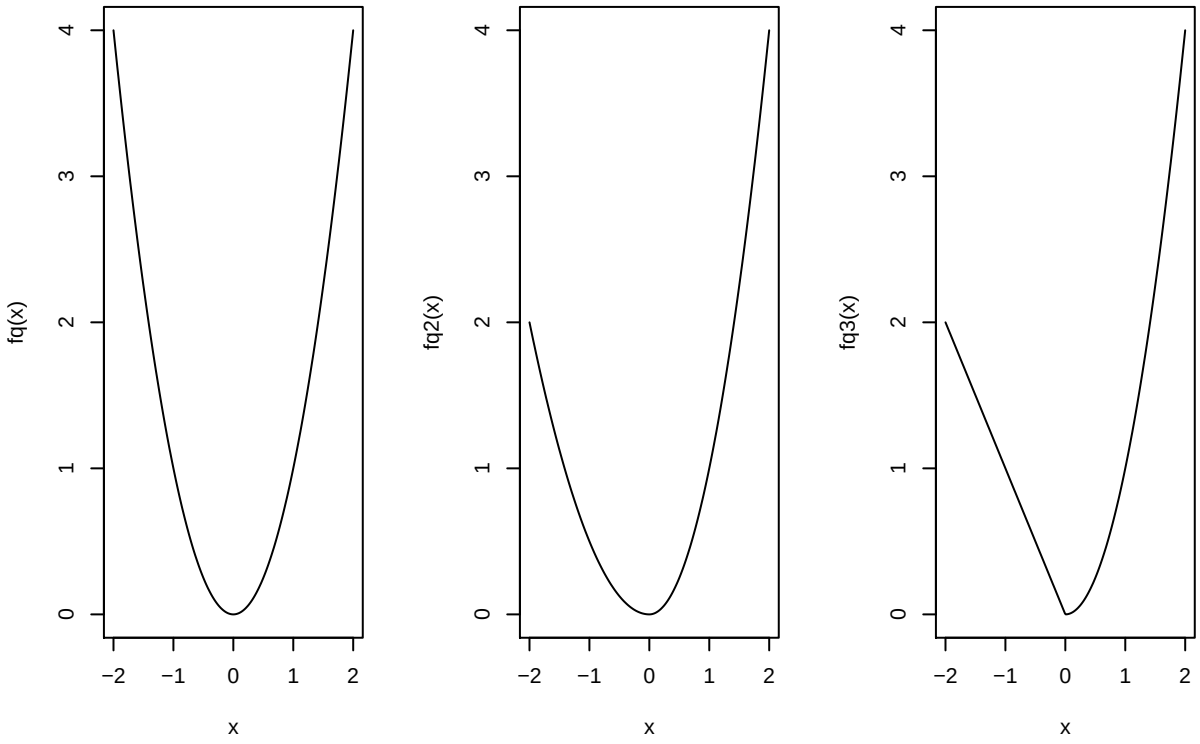
It shows this is indeed a very hard prediction problem!

According to the mse, model 4 was the best.

The MSE is a symmetric error measure. Sometimes, positive errors can be considered more severe than negative errors or vice versa. Asymmetric criteria can then be used.

For instance

```
##### Weighted error functions

fq <- function(x) x^2

fq2 <- function(x) 0.5^(x<0)*(x^2)

fq3 <- function(x) (x<0)*abs(x) + (x>=0)*x^2

par(mfrow = c(1, 3))

curve(fq, from = -2, to = 2, n = 300)

curve(fq2, from = -2, to = 2, n = 300)

curve(fq3, from = -2, to = 2, n = 300)
```



fq would correspond to the simple squared error, fq2 to the half of the unsquared error for negative residuals and fq3 for the raw residual for negative residuals.

```
res.fq2 <- colMeans(apply(cbind(yp.1.test,yp.2.test,yp.3.test,yp.4.test,yp.5.test)-test$y, 2, fq2))
names(res.fq2) <- c("mod1","mod2","mod3","mod4","mod5")
cat('results for error function fq2\n\n')
```

```
## results for error function fq2
```

```
print(res.fq2)
```

```
##   mod1   mod2   mod3   mod4   mod5
## 0.0537 0.0545 0.0551 0.0538 0.0548
```

```
cat('Best model is ',names(which.min(res.fq2)),'\n\n')
```

```
## Best model is  mod1
```

```
res.fq3 <- colMeans(apply(cbind(yp.1.test,yp.2.test,yp.3.test,yp.4.test,yp.5.test)-test$y, 2, fq3))
names(res.fq3) <- c("mod1","mod2","mod3","mod4","mod5")
cat('results for error function fq3\n\n')
```

```
## results for error function fq3
```

```
print(res.fq3)
```

```
##  mod1  mod2  mod3  mod4  mod5
## 0.130 0.131 0.131 0.130 0.131
```

```
cat('\nBest model is ',names(which.min(res.fq3)),'\n\n')
```

```
##
## Best model is  mod4
```

We now save the workspace so we can start at this point in the following practical. It includes all objects created in your session.

```
save.image(file.path(path,"image.RData"))
```