Advanced Methods for Regression and Classification

Exercise 8

Bosse Behrens , st.id: 12347333

Firts we load the packages and data.

```
library(ROCit)
library(dplyr)
```

```
##
## Attache Paket: 'dplyr'

## Die folgenden Objekte sind maskiert von 'package:stats':
##
##     filter, lag

## Die folgenden Objekte sind maskiert von 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
data(Diabetes)
set.seed(12347333)
```

We delete the columns id (it has no meaning towards diabetes), the location since the model would not work on data that is from different locaiton and also ethical reasons, the measurements for the second blood pressure since most of the data was missing in those columns and glyhb since the response it directly made up from it.

```
Diabetes <- Diabetes %>% select(-id, -location, -bp.2s, -bp.2d, -glyhb)
Diabetes <- na.omit(Diabetes)
Diabetes$dtest <- ifelse(Diabetes$dtest == "+", 1, 0)
```

We seperate the classes.

```
Diabetes_plus <- Diabetes[Diabetes$dtest == "1", ]
Diabetes_minus <- Diabetes[Diabetes$dtest == "0", ]
```

Now we split into train and test data according to the specified rules.

```
n_plus <- nrow(Diabetes_plus)
n_minus <- nrow(Diabetes_minus)

train_plus <- sample(1:n_plus, round((3/4) * n_plus))
```

```
test_plus <-(1:n_plus)[-train_plus]
train_minus <- sample(1:n_minus, round((3/4) * n_minus))
test_minus <-(1:n_minus)[-train_minus]

train_data_plus <- Diabetes_plus[train_plus, ]
test_data_plus <- Diabetes_plus[test_plus, ]

train_data_minus <- Diabetes_minus[train_minus, ]
test_data_minus <- Diabetes_minus[test_minus, ]

train_data <- rbind(train_data_plus, train_data_minus)

test_data <- rbind(test_data_plus, test_data_minus)
```

Task 1

First we fit a logistic regression model.

```
log_model <- glm(dtest ~ ., data = train_data, family = binomial)
```

The problem we face it that is does not converge and might not be very stable.

```
test_probs <- predict(log_model, newdata = test_data, type = "response")

test_pred <- ifelse(test_probs > 0.5, 1, 0)

conf_matrix <- table(Predicted = test_pred, Actual = test_data$dtest)

print(conf_matrix)
```

```
##           Actual
## Predicted  0  1
##         0 78  8
##         1  0  6
```

```
misclass_rate <- 1 - sum(diag(conf_matrix)) / sum(conf_matrix)
cat("Misclassification Rate:", misclass_rate, "\n")
```

```
## Misclassification Rate: 0.08695652
```

As we observe the miscalssification rate is low, but also the classes are very unbalanced and with positive cases it is more inaccurate.

Task 2

We now perform the same with the cv.glmnet logistic regression.

```
library(glmnet)
```

```
## Lade nötiges Paket: Matrix
```

```
## Loaded glmnet 4.1-8
```

```
x_train <- model.matrix(~ . - dtest - 1, data = train_data)
x_test <- model.matrix(~ . - dtest - 1, data = test_data)

y_train <- as.numeric(as.character(train_data$dtest))
y_test <- as.numeric(as.character(test_data$dtest))

cv_model <- cv.glmnet(x_train, y_train, family = "binomial", type.measure = "class")
```

We predict again in the test data and get the results.

```
test_probs <- predict(cv_model, newx = x_test, s = "lambda.min", type = "response")

test_pred <- ifelse(test_probs > 0.5, 1, 0)

conf_matrix <- table(Predicted = test_pred, Actual = y_test)

print(conf_matrix)
```

```
##          Actual
## Predicted  0  1
##         0 78  7
##         1  0  7
```

```
misclass_rate <- 1 - sum(diag(conf_matrix)) / sum(conf_matrix)
cat("Misclassification Rate:", misclass_rate, "\n")
```

```
## Misclassification Rate: 0.07608696
```

As we can see this already works a bit better.

Task 3

part a and b anc c

After first tests with fitting the model we saw it had problems and was also taking long to compute, so we set k=5 for the smoothed parameters. We choose to smooth all numeric, non-factorial variables, because we simply do not have the expert knowledge on which relationships might be linear and which ones not.

```
library(mgcv)
```

```
## Lade nötiges Paket: nlme
```

```
##
## Attache Paket: 'nlme'
```

```
## Das folgende Objekt ist maskiert 'package:dplyr':
##
##     collapse

## This is mgcv 1.9-1. For overview type 'help("mgcv-package")'.
```

```
gam_model <- gam(dtest ~
                   s(chol,  k = 5) +
                   s(stab.glu,  k = 5) +
                   s(hdl,  k = 5) +
                   s(ratio,  k = 5) +
                   s(age,  k = 5) +
                   s(bmi,  k = 5) +
                   s(whr,  k = 5) +
                   s(height,  k = 5) +
                   s(weight,  k = 5) +
                   s(bp.1s,  k = 5) +
                   s(bp.1d,  k = 5) +
                   s(time.ppn,  k = 5) +
                   gender +
                   frame,
                 data = train_data,
                 family = binomial)
summary(gam_model)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## dtest ~ s(chol, k = 5) + s(stab.glu, k = 5) + s(hdl, k = 5) +
##     s(ratio, k = 5) + s(age, k = 5) + s(bmi, k = 5) + s(whr,
##     k = 5) + s(height, k = 5) + s(weight, k = 5) + s(bp.1s, k = 5) +
##     s(bp.1d, k = 5) + s(time.ppn, k = 5) + gender + frame
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.8583     1.1482  -4.231 2.32e-05 ***
## gendermale   -0.2115     1.0140  -0.209    0.835
## framemedium   1.1434     0.8136   1.405    0.160
## framesmall    0.6009     1.2190   0.493    0.622
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df Chi.sq  p-value
## s(chol)     3.719  3.939 13.378  0.00904 **
## s(stab.glu) 2.657  3.157 32.365 4.74e-07 ***
## s(hdl)      1.000  1.000  3.613  0.05734 .
## s(ratio)    1.064  1.118  2.751  0.10548
## s(age)      2.830  3.352  3.836  0.35478
## s(bmi)      1.000  1.000  0.332  0.56465
## s(whr)      1.001  1.002  0.380  0.53901
```

```
## s(height)   1.000  1.000  0.105  0.74617
## s(weight)   1.000  1.000  0.376  0.53967
## s(bp.1s)    1.890  2.342  2.122  0.41793
## s(bp.1d)    1.000  1.000  1.327  0.24947
## s(time.ppn) 3.806  3.973  9.286  0.05317 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =   0.72   Deviance explained = 70.2%
## UBRE = -0.55546  Scale est. = 1         n = 274
```

As we can observe, stab.glu is significant at the 0.001 level, the intercept and time.ppn at the 0.01 level and gender, chol, age and bp.1d at the 0.1 level. The smooth functions are linear (edf=1) for age, bmi, whr, height and weight, meaning there is probabyly no overly complex relationship with the response. for bp.1s it is at 1.4, stab.glu and bp.1d at around 2 and ratio, chol and time.ppn at or slightly over 3.5. The higher this nuber is, the more complex the non-linear relationship.

part d

We plot the explanatory variables we smoothed against their smoothed values.

```
dev.new()
plot(gam_model, page = 3, shade = TRUE, shade.col = "yellow", scale=0)
```

We can observe that for the less compelx predictors, like weight and height these liens are straight which implies as mentioned some lienar relationship. As the dimensions increase (edf) the predictors with greather values such as time.ppn show more coplex patterns that show in more wavy and not as ismple lines. These imply non-linear relationships. The yellow part are the confidence intervals, which are mostly low at the center, where most of the data is for most distributions of the opredictors, whereas at the edges they get wider since there is less data and the confidence intervals cannot be as reliable anymore.

part e

We again obtain the predictions on the test data and clauclate confusion matrix anb misclassification rate.

```
test_probs <- predict(gam_model, newdata = test_data, type = "response")

test_pred <- ifelse(test_probs > 0.5, 1, 0)

conf_matrix <- table(Predicted = test_pred, Actual = test_data$dtest)
print(conf_matrix)
```

```
##           Actual
## Predicted  0  1
##         0 74  4
##         1  4 10
```

```
misclass_rate <- 1 - sum(diag(conf_matrix)) / sum(conf_matrix)
cat("Misclassification Rate:", misclass_rate, "\n")
```

```
## Misclassification Rate: 0.08695652
```

As we can see compared to the simple logistic mdoels in Task 2 and 3 the results got significantly worse.

part f

We use a shrinkage smoother with thin plate regression spline smoothers that shrinks some coefficients towards zero.

```
gam_model <- gam(dtest ~
                   s(chol, bs = "ts", k = 5) +
                   s(stab.glu, bs = "ts", k = 5) +
                   s(hdl, bs = "ts", k = 5) +
                   s(ratio, bs = "ts", k = 5) +
                   s(age, bs = "ts", k = 5) +
                   s(bmi, bs = "ts", k = 5) +
                   s(whr, bs = "ts", k = 5) +
                   s(height, bs = "ts", k = 5) +
                   s(weight, bs = "ts", k = 5) +
                   s(bp.1s, bs = "ts", k = 5) +
                   s(bp.1d, bs = "ts", k = 5) +
                   s(time.ppn, bs = "ts", k = 5) +
                   gender +
                   frame,
                 data = train_data,
                 family = binomial)


summary(gam_model)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## dtest ~ s(chol, bs = "ts", k = 5) + s(stab.glu, bs = "ts", k = 5) +
##     s(hdl, bs = "ts", k = 5) + s(ratio, bs = "ts", k = 5) + s(age,
##     bs = "ts", k = 5) + s(bmi, bs = "ts", k = 5) + s(whr, bs = "ts",
##     k = 5) + s(height, bs = "ts", k = 5) + s(weight, bs = "ts",
##     k = 5) + s(bp.1s, bs = "ts", k = 5) + s(bp.1d, bs = "ts",
##     k = 5) + s(time.ppn, bs = "ts", k = 5) + gender + frame
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.0381     0.8469  -4.768 1.86e-06 ***
## gendermale   -0.3577     0.6665  -0.537    0.591
## framemedium   0.8280     0.6779   1.221    0.222
## framesmall    0.4118     0.9534   0.432    0.666
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                   edf Ref.df Chi.sq p-value
## s(chol)     3.481e+00      4  9.556  0.0239 *
## s(stab.glu) 2.522e+00      4 39.883  <2e-16 ***
## s(hdl)      1.968e+00      4  2.908  0.1983
## s(ratio)    6.056e-05      4  0.000  0.2774
## s(age)      1.358e-05      4  0.000  0.4072
```

```
## s(bmi)      2.714e-06      4  0.000  0.8743
## s(whr)      9.449e-05      4  0.000  0.5222
## s(height)   3.133e-06      4  0.000  0.7870
## s(weight)   5.822e-06      4  0.000  0.7606
## s(bp.1s)    4.862e-05      4  0.000  0.3759
## s(bp.1d)    8.703e-01      4  4.227  0.0240 *
## s(time.ppn) 3.631e+00      4  9.540  0.0308 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =   0.68   Deviance explained = 65.5%
## UBRE = -0.58383  Scale est. = 1         n = 274
```

As we can see ratio, age, whr, height, weight and bp.1s are very small and close to zero. Again stab.glu is by far the most significant predictor. These predictors are still used but effectivvely not since they are so small. We now again predict on the test ste and plot confusion matrix and misclassification rate.

```r
test_probs <- predict(gam_model, newdata = test_data, type = "response")

test_pred <- ifelse(test_probs > 0.5, 1, 0)

conf_matrix <- table(Predicted = test_pred, Actual = test_data$dtest)
print(conf_matrix)
```

```
##          Actual
## Predicted  0  1
##         0 76  6
##         1  2  8
```

```r
misclass_rate <- 1 - sum(diag(conf_matrix)) / sum(conf_matrix)
cat("Misclassification Rate:", misclass_rate, "\n")
```

```
## Misclassification Rate: 0.08695652
```

We can observe a very slight improve compared to the normal gam we used before.

part g

We now train a GAM model again by now completely excluding all predictors that were shrunk very close to zero in part f.

```r
gam_model <- gam(dtest ~
                   s(chol, k = 5) +
                   s(stab.glu, k = 5) +
                   s(hdl, k = 5) +
                   s(bp.1d, k = 5) +
                   s(time.ppn, k = 5) +
                   gender +
                   frame,
                 data = train_data,
                 family = binomial)

summary(gam_model)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## dtest ~ s(chol, k = 5) + s(stab.glu, k = 5) + s(hdl, k = 5) +
##     s(bp.1d, k = 5) + s(time.ppn, k = 5) + gender + frame
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.0898     0.8580  -4.766 1.88e-06 ***
## gendermale   -0.4268     0.6796  -0.628    0.530
## framemedium   0.8484     0.6826   1.243    0.214
## framesmall    0.4239     0.9564   0.443    0.658
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df Chi.sq p-value
## s(chol)     3.469  3.828  9.582  0.0449 *
## s(stab.glu) 2.491  2.953 38.140  <2e-16 ***
## s(hdl)      1.968  2.443  2.398  0.3254
## s(bp.1d)    1.000  1.000  4.847  0.0277 *
## s(time.ppn) 3.626  3.912  9.587  0.0448 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.683   Deviance explained = 65.5%
## UBRE = -0.58392  Scale est. = 1         n = 274
```

We see that stab.glu is still by far the most significant predictor. The intercept is also very significant. gender, chol, hdl and time.ppn are significant at a higher level. Only frame and bp.1d are not showing as significant at all. Of the smoothed predicotrs only hdl shows a linear relationship while the other are more complex with higher edf values. The R squared (adjusted) is at 0.753 and therefore alright but not exceedingly satisfying. We again predict on tghe test data and get the confusion matrix and miscalssiication rate.

```
test_probs <- predict(gam_model, newdata = test_data, type = "response")

test_pred <- ifelse(test_probs > 0.5, 1, 0)

conf_matrix <- table(Predicted = test_pred, Actual = test_data$dtest)
print(conf_matrix)
```

```
##          Actual
## Predicted  0  1
##         0 76  6
##         1  2  8
```

```
misclass_rate <- 1 - sum(diag(conf_matrix)) / sum(conf_matrix)
cat("Misclassification Rate:", misclass_rate, "\n")
```

```
## Misclassification Rate: 0.08695652
```

As we can see this is performing the same as the one before. It is still performing worse than the Logistic models from task 1 and 2 though. Some more indepth analysis of which perdictors to smooth, which to select, etc. would be required to possibly improve a GAM mode further.