

Supplementary Material:

Table: Selection of tuning parameters using AUC as loss function in 10-fold cross validation repeated five times

Method	Parameter 1	Parameter 2	Parameter 3	Parameter 4
LOGIT	Default	-	-	-
LOGIT [†]	Default	-	-	-
RCS	Degree=3	-	-	-
MFP	Degree=4	-	-	-
RPART	Cp=0.0001 0 to 0.01 step by 0.0001	-	-	-
RF	TuneLength=10	n.trees=1000	-	-
GBM	n.trees=900 (100 to 2000 step by 100)	Interaction.depth=5 (1,3,5,7)	Shrinkage=0.01 (0.001,0.01,0.1)	n.minobsinnode=16 (1,6,11,16)
SVM	Cost=6 (0.1,0.5,1,2,4,6,8,10)	Gamma=0.1 (0.01,0.05 0.1,0.2,0.5,1)	-	-
NNET	Decay=0.102 (0 to 0.2 step by 0.004)	Size =1 (1,3,5)	-	-

R code:

```
#####
# Modelling - LOGIT
#####

modFormula <- paste("died.train~male +age + NEWS + ALB + CRE + HB + POT + SOD + WBC + URE ")
modFormula <- as.formula(modFormula)

model_logit = glm(modFormula, data = training, family = "binomial", x=TRUE, y=TRUE)

#####
# Modelling - LOGIT†
#####

modFormulat <- paste("died.train~male +age + NEWS + ALB + sqrt_CRE + HB + log_POT + SOD + log_WBC +
log_URE")
modFormulat <- as.formula(modFormulat)

model_logit_trans = glm(modFormulat, data = training, family = "binomial", x=TRUE, y=TRUE)

#####
# Modelling - RCS
#####

modFormularcs <- paste("died.train~male +rcs(age,3) + NEWS + rcs(ALB,3) + rcs(CRE,3) + rcs(HB,3) +
rcs(POT,3) + rcs(SOD,3) + rcs(WBC,3) + rcs(URE,3)")
modFormularcs <- as.formula(modFormularcs)

model_rcs = glm(modFormularcs,family = "binomial",data = training, x=TRUE, y=TRUE)

summary(model_rcs)

#####
# Modelling - MFP
#####

modFormulamfp <- paste("died.train~male +fp(age) + NEWS + fp(ALB) + fp(CRE) + fp(HB) + fp(POT) +
fp(SOD) + fp(WBC) + fp(URE)")
modFormulamfp <- as.formula(modFormulamfp)

model_mfp = mfp(modFormulamfp,family = "binomial",data = training, x=TRUE, y=TRUE)

#####
# Modelling - RPART
#####
#RPART
set.seed(669)

ctrl <- trainControl(method = "repeatedcv",repeats = 5,number = 10,classProbs = T,summaryFunction =
twoClassSummary)

rpartModel <- train(modFormula,
                    data = training,
                    method = "rpart",
                    tuneGrid =expand.grid(.cp=seq(0,0.01,length=100)),
                    metric="ROC",
                    trControl = ctrl
)

#####
# Modelling - RF
#####
#rf
set.seed(669)
rfModel <- train(modFormula,
                 data = training,
                 method = "rf",
                 tuneLength = 10,
                 ntrees = 1000,
```

```

        importance = TRUE,
        metric="ROC",
        trControl = ctrl
    )

#####
# Modelling - GBM
#####
#gbm
gbmGrid <- expand.grid(.interaction.depth = seq(1,7,by=2), .n.trees = seq(100, 2000, by =
100),.shrinkage = c(0.001,0.01, 0.1),.n.minobsinnode=seq(1,20,by=5))

set.seed(669)
gbmModel <- train(modFormula,
                  data = training,
                  method = "gbm",
                  tuneGrid = gbmGrid,
                  verbose = FALSE,
                  metric="ROC",
                  trControl = ctrl
                )

#####
# Modelling - SVM
#####
#SVM
ctrl <- trainControl(method = "repeatedcv",repeats = 5,number = 10,classProbs = T,summaryFunction =
twoClassSummary)

set.seed(669)
#creation of weights - also fast for very big datasets
weights <- as.numeric(died.train)-1

for(val in unique(weights)) {weights[weights==val]=1/sum(weights==val)*length(weights)/2} # normalized
to sum to length(samples)

svmModel <- train(modFormula,method = 'svmLinear',
                  maximize = T,
                  weights=weights,
                  tuneGrid=expand.grid(.C=c(0.1,0.5,1,2,4,6,8,10),.sigma=c(0.01,0.05,0.1,0.2,0.5,1)),
                  preProcess = c('center', 'scale'),
                  maxit=10000,
                  data=training,
                  metric="ROC",
                  trControl = ctrl
                )

#####
# Modelling - NNET
#####
#NNET
nnetGrid <- expand.grid(.decay = seq(0,0.2,length=50), .size =c(1,3,5))

set.seed(669)
nnetModel <- train(modFormula,
                  data = training,
                  method = "nnet",
                  tuneGrid = nnetGrid,
                  preProc = c("center", "scale"),
                  maxit=10000,
                  trace=F,
                  metric="ROC",
                  trControl = ctrl
                )

```