MRSA Klassifizierung in Schweinemastherden mit *glm*

Dr. Robert Opitz

2021-08-26

Der MRSA-Datensatz in Schweinemastherden wurde von Fromm *et al.* in PVM, **117**, 2014 veröffentlicht. Es handelt sich dabei um eine Metastudie. Diese wurde mittels logistischer Regression und generalized estiamtion equations ausgewertet. Es wurde eine Variablenselektion durch geführt: (i) durch univariates filtern (mit logistischer Regression) wurden alle Variablen entfernt, die keinen direkten Einfluss auf die Zielvariable haben, (ii) von paarweise korrelierten Variablen wurde eine der korrelierten Variablen entfernt. Es wurde versucht auch Interaktionsterme zubreücksichtigen. Es wurde nur einer gefunden, auch aufgrund der Anzahl der Datenpunkte Datensatzes. Dieser wurde im finalen Modell nicht weiter berücksichtigt.

# Notwendige Pakte und eigene Funktionen

Hier verwendete Pakete.

library(magrittr)  
library(caret)

Einige Hilfsfunktionen zum Plotten usw..

plot\_func <- function(model\_fit\_response, reference, title = "")   
{  
 lev <- levels(reference)  
 neg <- lev[1]; pos <- lev[2] # negative and positive class label  
   
 # compute accuracy, sensitivity, specificity, balancec accuracy   
 # for all thresholds  
 threshold <- seq(0, 1, 0.01)  
 n <- length(threshold)  
 acc <- rep(NA, n)  
 sen <- rep(NA, n)  
 spe <- rep(NA, n)  
 bacc <- rep(NA, n)  
 for (i in seq(n))   
 {  
 pred\_class <- factor(ifelse(model\_fit\_response <= threshold[i],   
 neg, pos),  
 levels = lev)  
 acc[i] <- mean(reference == pred\_class)  
 sen[i] <- mean(reference[reference == pos] == pred\_class[reference == pos])  
 spe[i] <- mean(reference[reference != pos] == pred\_class[reference != pos])  
 bacc[i] <- 0.5 \* (sen[i] + spe[i])  
 }  
  
 # dependece of Acc, Sen, Spe, and bal. Acc of Threshold;  
 # Threshold dependence  
 plot(threshold, acc,   
 type = "l",  
 xlim = c(0,1),  
 ylim = c(0,1),  
 ylab = "Acc/Sen/Spe/bal. Acc",  
 main = title)  
 lines(threshold, sen, col = "red")  
 lines(threshold, spe, col = "blue")  
 lines(threshold, bacc, col = "green")  
 abline(v = 0.5, lty = 4)  
 cat("Max. Acc = ", max(acc),   
 " at Threshold = ", threshold[which.max(acc)], "\n")  
 abline(v = thres <- threshold[which.max(bacc)], lty = 2)  
 cat("Max. bal. Acc = ", max(bacc), " at Threshold = ", thres, "\n")  
 # no information criteria  
 tmp <- summary(reference)  
 no\_info <- max(tmp) / sum(tmp)  
 abline(h = no\_info, lty = 3)  
  
 # Receiver Operating Curve  
 plot(1 - spe, sen,   
 type = "l",   
 main = paste("ROC", title),   
 lwd = 5)  
 abline(a = 0, b = 1, lty = 2)  
 abline(a = 1, b = -1, lty = 3)  
  
 print(pROC::roc(reference, model\_fit\_response))  
  
 # Youden's J curve  
 plot(1 - spe, sen + spe - 1,  
 ylim = c(0,1),  
 type = "l",  
 ylab = "J = sen + spe - 1",  
 main = paste("Youden's J", title),   
 lwd = 5)  
 abline(a = 1, b = -1, lty = 2)  
 max\_youden\_j <- which.max(sen + spe - 1)  
 cat("max. Youden's J = ", sen[max\_youden\_j] + spe[max\_youden\_j] - 1, " ",  
 "Sen = ", sen[max\_youden\_j], ";",  
 "Spe = ", spe[max\_youden\_j], ";",  
 "threshold = ", threshold[max\_youden\_j], "\n")  
 abline(v = 1 - spe[max\_youden\_j], lty = 3)  
   
 return(thres)  
}  
  
get\_and\_plot\_feature\_frequencies <- function(feature\_list,   
 my\_formula,   
 input\_data)   
{  
 term\_list <- vector( mode = 'list', length = length(feature\_list) )  
 all\_term\_names <- colnames( model.matrix( my\_formula,   
 data = input\_data ) )  
  
 for (i in seq\_along(term\_list))  
 term\_list[[i]] <- which( all\_term\_names %in% feature\_list[[i]] )  
  
 test <- rep( 0, length( all\_term\_names ) )  
 for (x in term\_list)  
 test[ x ] <- test[ x ] + 1  
   
 plot( sort(test/length(term\_list), decreasing = TRUE),  
 type = 'h',  
 xlab = 'term',   
 ylab = 'frequency')  
  
 abline( h = 0.9, lty = 2 )  
  
 df\_result <- data.frame(term\_counts = test,   
 frequency = test/length(term\_list),   
 term\_name = all\_term\_names)  
 df\_result <- df\_result[order(df\_result$frequency,   
 decreasing = TRUE),]  
  
 return(df\_result)  
}  
  
plot\_weights\_of\_feature <- function(weights\_list, feature\_name){  
 weights <- sapply(weights\_list,   
 function(x, feature\_name) x[feature\_name,], feature\_name)  
 hist(weights,  
 main = paste("Weight distribution for", feature\_name),  
 breaks = 100,  
 probability = TRUE)  
 invisible(NULL)  
}

# Datenvorbereitung

## Einlesen und Vorbereiten der Daten.

Einlesen und Bereinigung der Daten. Es wird sichergestellt, dass alle Daten einen R-konformen Namen haben und als Faktoren vorliegen. die Variablennamen erhalten die Namen aus dem Paper. Einige der Kategorien der Variablen werden besser lesbar gestaltet.

MRSA\_schweineherden <- read.csv(file = file.choose(),  
 header = TRUE)  
  
# all categorigal variables needs to be checked if there name is valid in R  
# E. g.: TRUE and FALSE are not; TRUE. and FALSE. are  
# All categorical Variables need to be (at least) described as factors  
for (i in seq\_len(ncol(MRSA\_schweineherden)))   
{  
 MRSA\_schweineherden[[i]] <- make.names(MRSA\_schweineherden[[i]])  
 MRSA\_schweineherden[[i]] <- factor(MRSA\_schweineherden[[i]])  
}  
  
# the old names in the data set are renamed, so they match the names in the paper  
new\_col\_names <- list(HerdMRSA = 'HERD\_MRSA',  
 HerdTypeNum = 'HERD\_TYPE',  
 FeedingPlacesMinGrouped = 'HERD\_SIZE',  
 PurchaseBin = 'PURCHASE',  
 AntibioticsFrom10W = 'AM\_DRUG',  
 AllInAllOut = 'ALL\_IN\_or\_OUT',  
 Cleanup = 'CLEAN\_UP',  
 Disinfection = 'DISINFECTION',  
 SlattedFloor = 'SLATTED',  
 OrganicFarm = 'ORGANIC',  
 OtherLivestockAtFarmBin = 'OTHER\_LIVESTOCK',  
 CompanionAnimalsBin = 'COMPANION',  
 IndoorHousing = 'INDOOR' )  
  
# rename the column names  
column\_names <- colnames(MRSA\_schweineherden)  
for (col\_name in column\_names)  
 colnames(MRSA\_schweineherden)[column\_names == col\_name] <- new\_col\_names[[col\_name]]  
  
# replace old variable types for variable HERD\_SIZE  
levels(MRSA\_schweineherden$HERD\_SIZE)

## [1] "X..5000.pigs" "X0.499.pigs" "X1000.4999.pigs" "X500.999.pigs"

levels(MRSA\_schweineherden$HERD\_SIZE) <- c('huge', 'small', 'large', 'medium')  
# reorder factors  
MRSA\_schweineherden$HERD\_SIZE <- factor(MRSA\_schweineherden$HERD\_SIZE,   
 levels = c('small', 'medium',   
 'large', 'huge'),  
 ordered = TRUE)  
levels(MRSA\_schweineherden$HERD\_SIZE)

## [1] "small" "medium" "large" "huge"

levels(MRSA\_schweineherden$HERD\_TYPE)

## [1] "farrow.to.finisher" "grower.to.finisher" "weaner.to.finisher"

levels(MRSA\_schweineherden$HERD\_TYPE) <- c("farrow", "grower", "weaner")  
levels(MRSA\_schweineherden$HERD\_TYPE)

## [1] "farrow" "grower" "weaner"

# replace old variable types for variable SLATTED\_AT\_LEAST\_PARTIALLY  
levels(MRSA\_schweineherden$SLATTED)

## [1] "no..not.slatted." "yes..at.least.partially.slatted."

levels(MRSA\_schweineherden$SLATTED) <- c('no', 'yes')  
levels(MRSA\_schweineherden$SLATTED)

## [1] "no" "yes"

## Vorstellung des Datensatzes (EDA)

str(MRSA\_schweineherden)

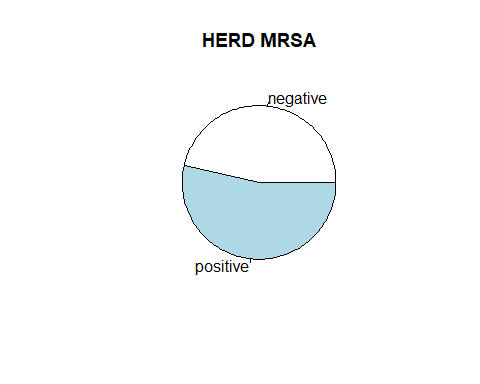
## 'data.frame': 400 obs. of 13 variables:  
## $ HERD\_MRSA : Factor w/ 2 levels "negative","positive": 2 2 1 2 1 2 2 2 2 2 ...  
## $ HERD\_TYPE : Factor w/ 3 levels "farrow","grower",..: 2 2 2 1 1 1 2 2 2 2 ...  
## $ HERD\_SIZE : Ord.factor w/ 4 levels "small"<"medium"<..: 3 3 2 1 2 2 2 2 3 3 ...  
## $ AM\_DRUG : Factor w/ 2 levels "no","yes": 2 2 1 1 1 1 2 1 2 1 ...  
## $ PURCHASE : Factor w/ 2 levels "no","yes": 2 2 2 1 1 1 2 2 2 2 ...  
## $ INDOOR : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...  
## $ OTHER\_LIVESTOCK: Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 2 2 1 1 ...  
## $ COMPANION : Factor w/ 2 levels "no","yes": 1 1 1 2 2 2 1 2 1 2 ...  
## $ ALL\_IN\_or\_OUT : Factor w/ 2 levels "no","yes": 2 2 1 1 1 2 2 1 2 2 ...  
## $ CLEAN\_UP : Factor w/ 2 levels "no","yes": 2 2 2 1 2 2 2 2 2 2 ...  
## $ DISINFECTION : Factor w/ 2 levels "no","yes": 2 2 2 1 2 2 2 2 2 2 ...  
## $ SLATTED : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...  
## $ ORGANIC : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...

Bestimmung der Anzahl der Datenpunkte, Anzahl Variablen und Verteilung der Response.

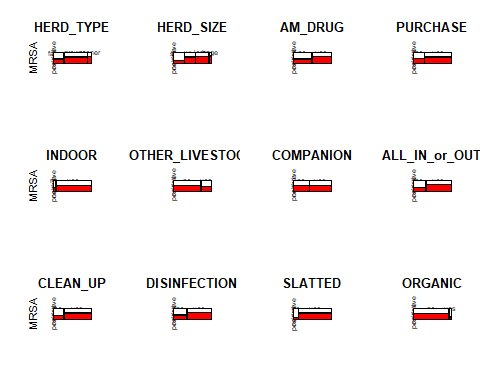
MRSA\_schweineherden$HERD\_MRSA %>% length -> number\_of\_datapoints  
MRSA\_schweineherden[1,-1] %>% length -> number\_of\_variables  
MRSA\_schweineherden$HERD\_MRSA %>% summary

## negative positive   
## 186 214

MRSA\_schweineherden$HERD\_MRSA %>% summary %>% pie(.,main = 'HERD MRSA')



Datenpunkte=400, Variablen=12.



nearZeroVar(MRSA\_schweineherden, saveMetrics = TRUE)

## freqRatio percentUnique zeroVar nzv  
## HERD\_MRSA 1.150538 0.50 FALSE FALSE  
## HERD\_TYPE 2.252252 0.75 FALSE FALSE  
## HERD\_SIZE 1.247863 1.00 FALSE FALSE  
## AM\_DRUG 1.105263 0.50 FALSE FALSE  
## PURCHASE 2.603604 0.50 FALSE FALSE  
## INDOOR 20.052632 0.50 FALSE TRUE  
## OTHER\_LIVESTOCK 2.738318 0.50 FALSE FALSE  
## COMPANION 1.409639 0.50 FALSE FALSE  
## ALL\_IN\_or\_OUT 2.149606 0.50 FALSE FALSE  
## CLEAN\_UP 2.846154 0.50 FALSE FALSE  
## DISINFECTION 1.941176 0.50 FALSE FALSE  
## SLATTED 7.695652 0.50 FALSE FALSE  
## ORGANIC 16.391304 0.50 FALSE FALSE

Die Variable INDOOR besteht fast ausschließlich aus Herden, die indoor leben, fast keine die outdoor sind. Dadurch haben wir eine near-zero-variance (nzv) für diese Variable. Wir entfernen diese Variable daher.

MRSA\_schweineherden\_reduced <- MRSA\_schweineherden  
MRSA\_schweineherden\_reduced$INDOOR <- NULL

# Erstes Modell

first\_model <- glm(HERD\_MRSA ~ .,  
 data = MRSA\_schweineherden\_reduced,  
 family = binomial(link = "logit"))  
  
pred\_prob <- predict(first\_model, type = "response")  
pred\_class <- ifelse(pred\_prob <= 0.54, "negative", "positive")  
table(pred\_class)

## pred\_class  
## negative positive   
## 167 233

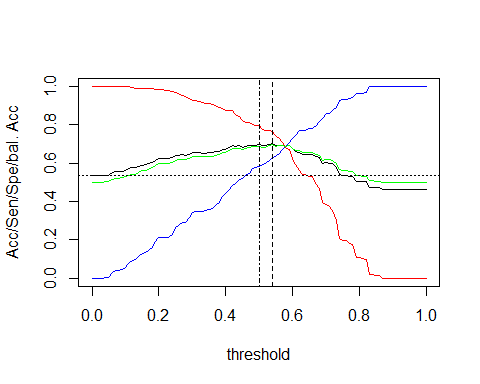
table(MRSA\_schweineherden\_reduced$HERD\_MRSA)

##   
## negative positive   
## 186 214

confusionMatrix(data = factor(pred\_class),  
 reference = MRSA\_schweineherden\_reduced$HERD\_MRSA,  
 positive = "positive")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction negative positive  
## negative 117 50  
## positive 69 164  
##   
## Accuracy : 0.7025   
## 95% CI : (0.6551, 0.7469)  
## No Information Rate : 0.535   
## P-Value [Acc > NIR] : 5.827e-12   
##   
## Kappa : 0.398   
##   
## Mcnemar's Test P-Value : 0.09893   
##   
## Sensitivity : 0.7664   
## Specificity : 0.6290   
## Pos Pred Value : 0.7039   
## Neg Pred Value : 0.7006   
## Prevalence : 0.5350   
## Detection Rate : 0.4100   
## Detection Prevalence : 0.5825   
## Balanced Accuracy : 0.6977   
##   
## 'Positive' Class : positive   
##

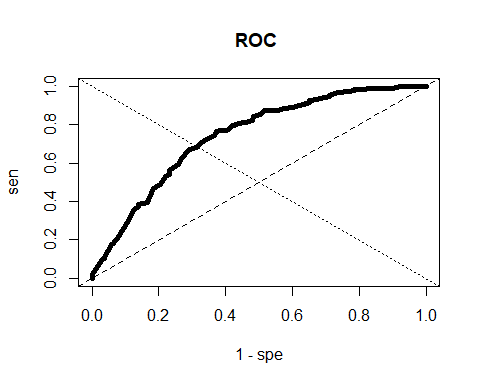
plot\_func(pred\_prob, MRSA\_schweineherden\_reduced$HERD\_MRSA)



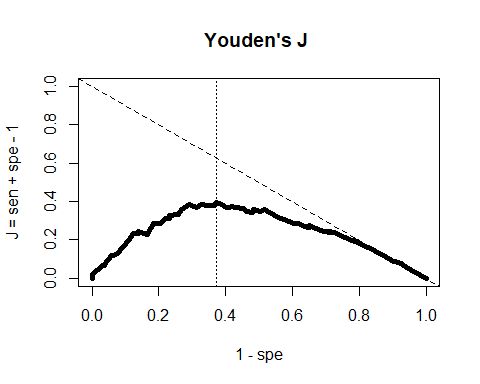
## Max. Acc = 0.7025 at Threshold = 0.54   
## Max. bal. Acc = 0.6976937 at Threshold = 0.54

## Setting levels: control = negative, case = positive

## Setting direction: controls < cases



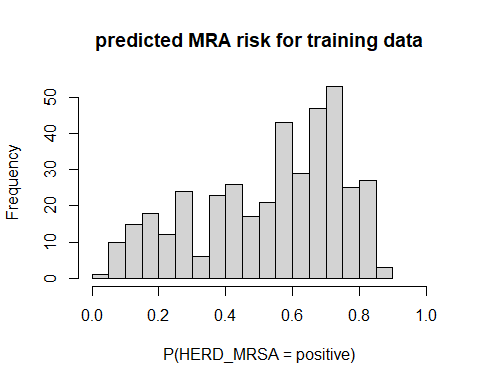
##   
## Call:  
## roc.default(response = reference, predictor = model\_fit\_response)  
##   
## Data: model\_fit\_response in 186 controls (reference negative) < 214 cases (reference positive).  
## Area under the curve: 0.7413



## max. Youden's J = 0.3953874 Sen = 0.7663551 ; Spe = 0.6290323 ; threshold = 0.54

## [1] 0.54

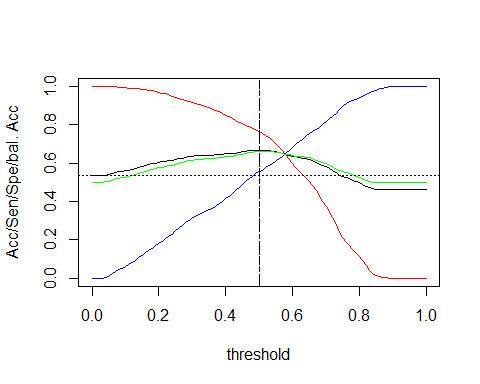
hist(pred\_prob,   
 xlim = c(0,1),   
 breaks = 25,  
 main = "predicted MRA risk for training data",  
 xlab = "P(HERD\_MRSA = positive)")



# Repeqted Crossvalidation

train\_index <- createMultiFolds(MRSA\_schweineherden\_reduced$HERD\_MRSA,  
 k = 10,  
 times = 100 )

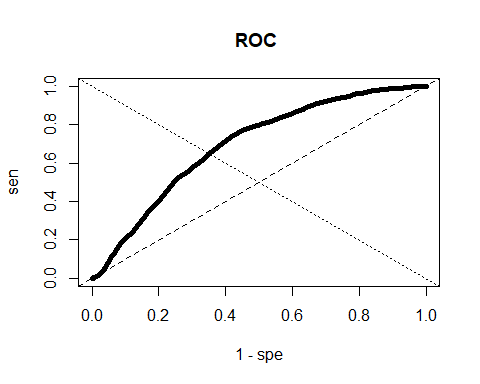
trained\_models <- lapply(train\_index,  
 function(this\_index)  
 {  
 df\_train <- MRSA\_schweineherden\_reduced[this\_index,]  
 trained\_model <- glm(HERD\_MRSA ~ .,  
 data = df\_train,  
 family = binomial(link = "logit"))  
 trained\_model  
 })  
# get predicted probabilities  
pred\_prob <- mapply(function(trained\_model, train\_data)  
 {  
 df\_test <- MRSA\_schweineherden\_reduced[-train\_data,]  
 predict(trained\_model,  
 newdata = df\_test,  
 type = "response")  
 }, trained\_models, train\_index)  
pred\_prob <- unlist(pred\_prob)  
# get references classes of test data  
ref\_class <- unlist(lapply(train\_index,   
 function(x) MRSA\_schweineherden\_reduced$HERD\_MRSA[-x]))  
# plot everything for test data  
plot\_func(pred\_prob, ref\_class)



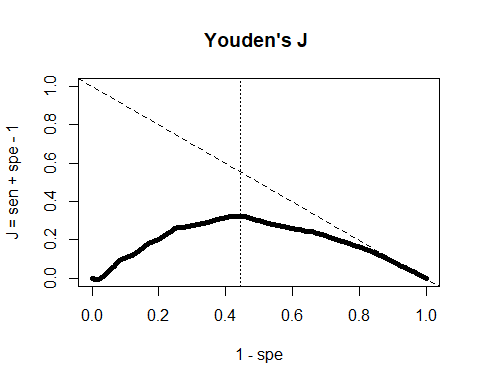
## Max. Acc = 0.668475 at Threshold = 0.5   
## Max. bal. Acc = 0.6610655 at Threshold = 0.5

## Setting levels: control = negative, case = positive

## Setting direction: controls < cases



##   
## Call:  
## roc.default(response = reference, predictor = model\_fit\_response)  
##   
## Data: model\_fit\_response in 18600 controls (reference negative) < 21400 cases (reference positive).  
## Area under the curve: 0.6941



## max. Youden's J = 0.3221309 Sen = 0.7669159 ; Spe = 0.5552151 ; threshold = 0.5

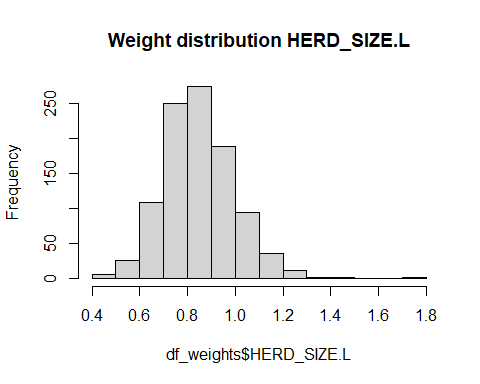
## [1] 0.5

# Distribution of weights in all training models

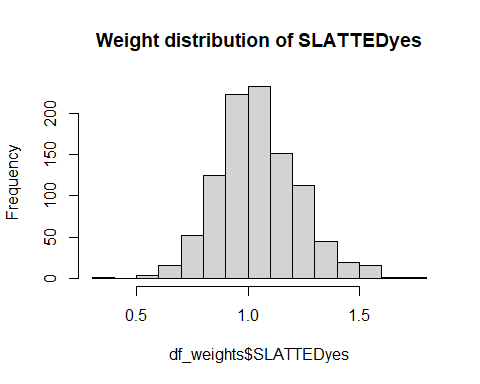
df\_weights <- NULL  
for (trained\_model in trained\_models)  
 df\_weights <- rbind.data.frame(df\_weights,  
 coef(trained\_model))  
colnames(df\_weights) <- names(coef(trained\_models[[1]]))

Plot some weight distributions

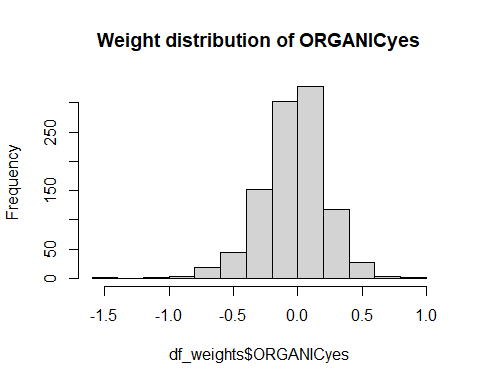
hist(df\_weights$HERD\_SIZE.L, main = "Weight distribution HERD\_SIZE.L")



hist(df\_weights$SLATTEDyes, main = "Weight distribution of SLATTEDyes")



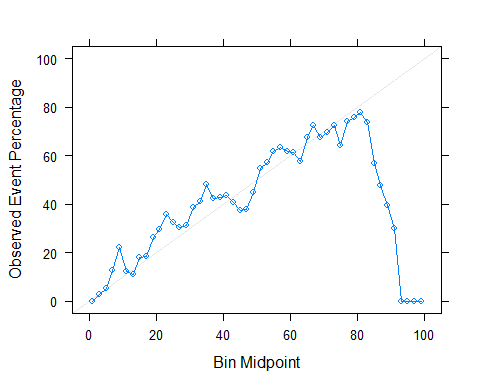
hist(df\_weights$ORGANICyes, main = "Weight distribution of ORGANICyes")



# Take a closer look – predicted distribution of risk, and its parts

Plot the probaility calibration curve

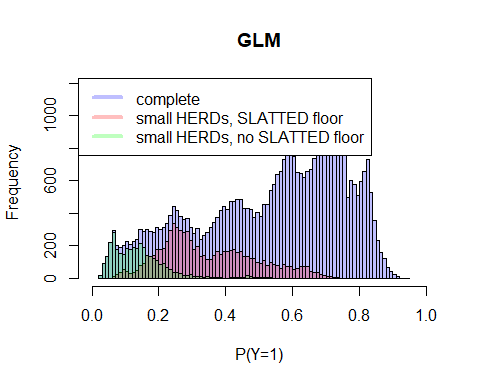
df\_cal <- data.frame(Class = ref\_class)  
df\_cal$glm <- pred\_prob  
cal\_obj <- caret::calibration(Class ~ glm,  
 data = df\_cal,  
 cuts = 50,  
 class = "positive")  
plot(cal\_obj)



Some helper functions to startify the predicted risk.

my\_subset <- function(df\_input, selection\_rule)  
{  
 selected\_col <- selection\_rule[1]  
 selected\_value <- selection\_rule[2]  
 df\_output <- NULL  
 for (i in seq\_len(nrow(df\_input)))  
 {  
 if (df\_input[i,selected\_col] == selected\_value)  
 df\_output <- rbind.data.frame(df\_output,  
 df\_input[i,])  
 }  
 if (!is.null(df\_output))  
 colnames(df\_output) <- colnames(df\_input)  
 df\_output  
}  
  
get\_risk\_subset <- function(df\_input, trained\_models,   
 train\_index, subset\_list)  
{  
 result <- mapply(function(trained\_model, train\_data)  
 {  
 df\_test <- df\_input[-train\_data,]  
 for (this\_subset in subset\_list)  
 {  
 df\_test <- my\_subset(df\_test, this\_subset)  
 if (is.null(df\_test))  
 return(NULL)  
 }  
 predict(trained\_model,  
 newdata = df\_test,  
 type = "response")  
 }, trained\_models, train\_index)  
 # remove empty entries in result list  
 result <- Filter(Negate(is.null), result)  
 unname(unlist(result)) # simple vector with predicted risks  
}

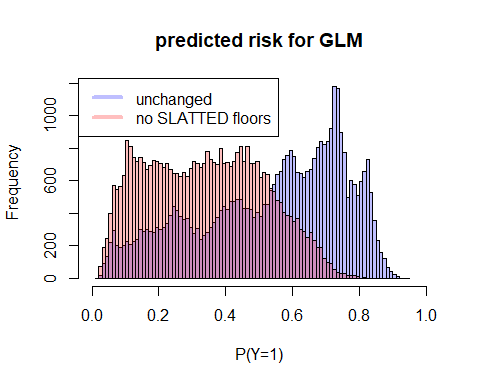
# original distribution of predicted risk  
org\_hist <- hist(pred\_prob,  
 breaks = -1\*Reduce("-", range(pred\_prob))/0.01,  
 plot = FALSE)  
  
sub\_dist\_A <- get\_risk\_subset(MRSA\_schweineherden\_reduced,  
 trained\_models, train\_index,  
 list(c("HERD\_SIZE", "small"),  
 c("SLATTED", "yes")))  
  
sub\_dist\_B <- get\_risk\_subset(MRSA\_schweineherden\_reduced,  
 trained\_models, train\_index,  
 list(c("HERD\_SIZE", "small"),  
 c("SLATTED", "no")))  
  
# distribution of stratified sample  
sub\_dist\_A\_hist <- hist(sub\_dist\_A,  
 breaks = -1\*Reduce("-", range(sub\_dist\_A))/0.01,  
 plot = FALSE)  
sub\_dist\_B\_hist <- hist(sub\_dist\_B,  
 breaks = -1\*Reduce("-", range(sub\_dist\_B))/0.01,  
 plot = FALSE)  
# plot it all  
plot(org\_hist,  
 freq = TRUE,  
 col = rgb(0,0,1,1/4),  
 xlim = c(0,1),  
 xlab = "P(Y=1)",  
 main = "GLM")  
plot(sub\_dist\_A\_hist,  
 freq = TRUE,  
 add = TRUE,  
 col = rgb(1,0,0,1/4))  
plot(sub\_dist\_B\_hist,  
 freq = TRUE,  
 add = TRUE,  
 col = rgb(0,1,0,1/4))  
legend("topleft",  
 legend = c("complete",  
 "small HERDs, SLATTED floor",#"sub\_dist\_A",  
 "small HERDs, no SLATTED floor"),#"sub\_dist\_B"),  
 col = c(rgb(0,0,1,1/4),  
 rgb(1,0,0,1/4),  
 rgb(0,1,0,1/4)),  
 lty = 1, lwd = 4)



# Change in predicted risk distribution

get\_mod\_pred\_risk <- function(df\_input, trained\_models,   
 train\_index, subset\_list)  
{  
 col\_name <- subset\_list[1]  
 new\_value <- subset\_list[2]  
 result <- mapply(function(trained\_model, train\_data)  
 {  
 df\_test <- df\_input[-train\_data,]  
 lev <- levels(df\_test[[col\_name]])  
 df\_test[,col\_name] <- factor(rep(new\_value,   
 nrow(df\_test)),  
 levels = lev)  
 pred <- predict(trained\_model,  
 newdata = df\_test,  
 type = "response")  
 }, trained\_models, train\_index)  
 # remove empty entries in result list  
 result <- Filter(Negate(is.null), result)  
 unname(unlist(result)) # simple vector with predicted risks  
}

# original distribution of predicted risk  
org\_hist <- hist(pred\_prob,  
 breaks = -1\*Reduce("-", range(pred\_prob))/0.01,  
 plot = FALSE)  
  
sub\_dist\_A <- get\_mod\_pred\_risk(MRSA\_schweineherden\_reduced,  
 trained\_models, train\_index,  
 c("SLATTED", "no"))  
  
# distribution of stratified sample  
sub\_dist\_A\_hist <- hist(sub\_dist\_A,  
 breaks = -1\*Reduce("-", range(sub\_dist\_A))/0.01,  
 plot = FALSE)  
  
# plot it all  
plot(org\_hist,  
 freq = TRUE,  
 col = rgb(0,0,1,1/4),  
 xlim = c(0,1),  
 xlab = "P(Y=1)",  
 main = "predicted risk for GLM")  
plot(sub\_dist\_A\_hist,  
 freq = TRUE,  
 add = TRUE,  
 col = rgb(1,0,0,1/4))  
legend("topleft",  
 legend = c("unchanged",  
 "no SLATTED floors"),#"sub\_dist\_A",  
 col = c(rgb(0,0,1,1/4),  
 rgb(1,0,0,1/4)),  
 lty = 1, lwd = 4)

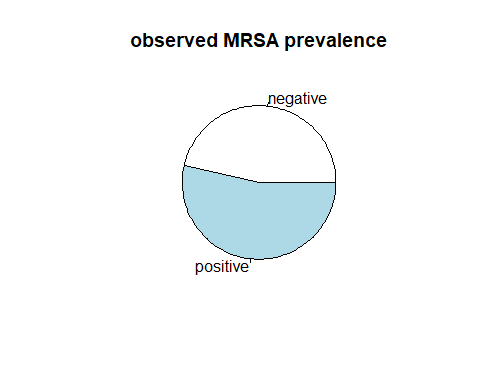


# Change in Predicted Prevalence

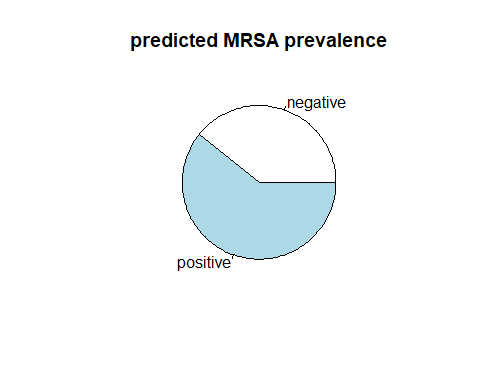
get\_mod\_pred\_prev <- function(df\_input, trained\_models,   
 train\_index, subset\_list, threshold)  
{  
 col\_name <- subset\_list[1]  
 new\_value <- subset\_list[2]  
 result <- mapply(function(trained\_model, train\_data)  
 {  
 df\_test <- df\_input[-train\_data,]  
 lev <- levels(df\_test[[col\_name]])  
 df\_test[,col\_name] <- factor(rep(new\_value,   
 nrow(df\_test)),  
 levels = lev)  
 pred <- predict(trained\_model,  
 newdata = df\_test,  
 type = "response")  
 ifelse(pred > threshold, 1, 0)  
 }, trained\_models, train\_index)  
 # remove empty entries in result list  
 result <- Filter(Negate(is.null), result)  
 unname(unlist(result)) # simple vector with predicted risks  
}

pred\_prev <- mapply(function(trained\_model, train\_data)  
 {  
 df\_test <- MRSA\_schweineherden\_reduced[-train\_data,]  
 pred <- predict(trained\_model,  
 newdata = df\_test,  
 type = "response")  
 ifelse(pred > 0.51, 1, 0)  
 }, trained\_models, train\_index)  
pred\_prev <- mean(unlist(pred\_prev))  
  
mod\_pred\_prev <- get\_mod\_pred\_prev(MRSA\_schweineherden\_reduced,   
 trained\_models,  
 train\_index,  
 c("SLATTED", "no"),  
 0.51)  
mod\_pred\_prev <- mean(mod\_pred\_prev)

obs\_prev <- mean(MRSA\_schweineherden\_reduced$HERD\_MRSA == "positive")  
pie(c(1 - obs\_prev, obs\_prev),  
 labels = c("negative", "positive"),  
 main = 'observed MRSA prevalence')



pie(c(1 - pred\_prev, pred\_prev),   
 labels = c("negative", "positive"),  
 main = 'predicted MRSA prevalence')



pie(c(1 - mod\_pred\_prev, mod\_pred\_prev),   
 labels = c("negative", "positive"),  
 main = 'pred. no SLATTED prevalence')

