#R code for a data-generating process based on random forests

# THIS CODE IS PROVIDED FOR ILLUSTRATIVE PURPOSES AND COMES WITH

# ABSOLUTELY NO WARRANTY.

# 1) Applies a random forest to the EFFECT1 dataset.

# 2) Determines the predicted probability of the event in EFFECT1 and EFFECT2

# data using the model fit to the EFFECT1 data.

# 3) For each subject in each of the two phases of EFFECT, a binary outcome is

# simulated using this predicted probability.

# 4) The other models are then applied to the EFFECT data with these simulated

# outcomes:

# 5) Each model is developed in the EFFECT1 sample (with simulated outcome).

# 6) Each model is then applied to the EFFECT2 sample

# (with the simulated outcome).

library(rms)

library(randomForest)

library(gbm)

library(glmnet)

################################################################################

# Fix number of trees for different methods

################################################################################

n.tree.rf <- 500

n.tree.bagg <- 500

# Default number of trees for RF/Bagging

n.tree.gbm <- 100

# Default number of trees for GBM.

################################################################################

# Read in parameter values from grid search for tuning ML parameters.

################################################################################

tune.bagg.list <- list(bagg.nodesize=0)

tune.bagg <- scan("TUNE/BAGG/bagg.optimal",tune.bagg.list)

tune.gbm.list <- list(gbm.interaction.depth=0,gbm.shrinkage=0)

tune.gbm <- scan("TUNE/GBM/gbm.optimal",tune.gbm.list)

tune.lrm.list <- list(lrm.knot=0)

tune.lrm <- scan("TUNE/LRM/lrm.optimal",tune.lrm.list)

tune.rf.list <- list(rf.mtry=0,rf.nodesize=0)

tune.rf <- scan("TUNE/RF/rf.optimal",tune.rf.list)

tune.lasso.list <- list(lasso.lambda=0)

tune.lasso <- scan("TUNE/LASSO2/lasso.optimal",tune.lasso.list)

tune.ridge.list <- list(ridge.lambda=0)

tune.ridge <- scan("TUNE/RIDGE2/ridge.optimal",tune.ridge.list)

remove(tune.bagg.list,tune.gbm.list,tune.lrm.list,tune.rf.list,

tune.lasso.list,tune.ridge.list)

################################################################################

# Reads in EFFECT1 and EFFECT2 data.

# Note: the data are not available for dissemination due to privacy regulations.

# Do not contact the author for the data.

################################################################################

effect1.df <- read.table("ami\_1.txt",header=T)

effect2.df <- read.table("ami\_2.txt",header=T)

################################################################################

# Create matrices for use with LASSO and ridge regression

################################################################################

attach(effect1.df)

X.derive <- cbind(age,female,cshock,acpulmed,sysbp,

diasbp,hrtrate,resp,diabetes,highbp,smokhx,dyslip,famhxcad,cvatia,

angina,cancer,dementia,pud,prevmi,asthma,depres,perartdis,prevrevasc,

chf,hyprthyr,as,hgb,wbc,sod,pot,glucose,urea,cr)

detach(effect1.df)

attach(effect2.df)

X.valid <- cbind(age,female,cshock,acpulmed,sysbp,

diasbp,hrtrate,resp,diabetes,highbp,smokhx,dyslip,famhxcad,cvatia,

angina,cancer,dementia,pud,prevmi,asthma,depres,perartdis,prevrevasc,

chf,hyprthyr,as,hgb,wbc,sod,pot,glucose,urea,cr)

detach(effect2.df)

################################################################################

# Fits random forest to the EFFECT1 sample.

# Apply fitted model to EFFECT1 and EFFECT2 samples to get predicted probabilities

# to be used to generate outcomes.

# For different data-generating processes, code for the appropriate model

# replaces the random forest code in this block of code.

################################################################################

set.seed(28042020)

rf.effect1 <- randomForest(as.factor(mort1yr) ~ age + female + cshock +

acpulmed + sysbp +

diasbp + hrtrate + resp + diabetes + highbp + smokhx + dyslip + famhxcad +

cvatia + angina + cancer + dementia + pud + prevmi + asthma + depres +

perartdis + prevrevasc + chf + hyprthyr + as + hgb + wbc + sod + pot +

glucose + urea + cr,

mtry=tune.rf$rf.mtry,nodesize=tune.rf$rf.nodesize,

ntree=n.tree.rf,replace=T,importance=T,

data=effect1.df)

prob.rf1 <- predict(rf.effect1,type="prob",newdata=effect1.df)[,2]

prob.rf2 <- predict(rf.effect1,type="prob",newdata=effect2.df)[,2]

effect1.df$prob.rf1 <- prob.rf1

effect2.df$prob.rf2 <- prob.rf2

################################################################################

# Main body of simulations.

################################################################################

for (iter in 1:1000){

set.seed(iter)

################################################################################

# Random forests data-generating process

# Generate outcomes in EFFECT1 and EFFECT2 using probabilities from RF.

################################################################################

effect1.df$Y <- rbinom(nrow(effect1.df),1,effect1.df$prob.rf1)

effect2.df$Y <- rbinom(nrow(effect2.df),1,effect2.df$prob.rf2)

################################################################################

# Apply logistic regression to simulated data.

################################################################################

lrm.1 <- lrm(Y ~ rcs(age,tune.lrm$lrm.knot) + female + cshock + acpulmed +

rcs(sysbp,tune.lrm$lrm.knot) + rcs(diasbp,tune.lrm$lrm.knot) +

rcs(hrtrate,tune.lrm$lrm.knot) + rcs(resp,tune.lrm$lrm.knot) + diabetes +

highbp + smokhx + dyslip + famhxcad + cvatia + angina + cancer + dementia +

pud + prevmi + asthma + depres + perartdis + prevrevasc + chf + hyprthyr +

as + rcs(hgb,tune.lrm$lrm.knot) + rcs(wbc,tune.lrm$lrm.knot) +

rcs(sod,tune.lrm$lrm.knot) + rcs(pot,tune.lrm$lrm.knot) +

rcs(glucose,tune.lrm$lrm.knot) + rcs(urea,tune.lrm$lrm.knot) +

rcs(cr,tune.lrm$lrm.knot),

data=effect1.df)

pred.valid.lrm.xbeta <- predict(lrm.1,newdata=effect2.df)

pred.valid.lrm <- exp(pred.valid.lrm.xbeta)/(1 + exp(pred.valid.lrm.xbeta))

val.lrm <- val.prob(pred.valid.lrm,effect2.df$Y,pl=F)

roc.valid.lrm <- val.lrm["C (ROC)"]

r2.valid.lrm <- val.lrm["R2"]

brier.valid.lrm <- val.lrm["Brier"]

ici.valid.lrm <- val.lrm["Eavg"]

E90.valid.lrm <- val.lrm["E90"]

Emax.valid.lrm <- val.lrm["Emax"]

intercept.valid.lrm <- val.lrm["Intercept"]

slope.valid.lrm <- val.lrm["Slope"]

p.valid <- mean(effect2.df$Y)

brier.max.valid <- p.valid\*((1-p.valid)^2) + (1-p.valid)\*(p.valid^2)

brier.max.valid.lrm <- 1 - (brier.valid.lrm/brier.max.valid)

pred.valid.lrm <- ifelse(pred.valid.lrm==0,0.00001,pred.valid.lrm)

negLL <- -sum((effect2.df$Y)\*log(pred.valid.lrm) +

(1-effect2.df$Y)\*log(1-pred.valid.lrm))

cat(iter,roc.valid.lrm,r2.valid.lrm,brier.valid.lrm,brier.max.valid.lrm,

ici.valid.lrm,E90.valid.lrm,Emax.valid.lrm,intercept.valid.lrm,

slope.valid.lrm,negLL,

file="dgp.rf.lrm.out",fill=T,append=T)

remove(lrm.1,pred.valid.lrm.xbeta,pred.valid.lrm,val.lrm)

################################################################################

# Apply random forests to the simulated data.

################################################################################

rf.1 <- randomForest(as.factor(Y) ~ age + female + cshock + acpulmed + sysbp +

diasbp + hrtrate + resp + diabetes + highbp + smokhx + dyslip + famhxcad +

cvatia + angina + cancer + dementia + pud + prevmi + asthma + depres +

perartdis + prevrevasc + chf + hyprthyr + as + hgb + wbc + sod + pot +

glucose + urea + cr,

mtry=tune.rf$rf.mtry,nodesize=tune.rf$rf.nodesize,

ntree=n.tree.rf,replace=T,importance=T,

data=effect1.df)

pred.valid.rf <- predict(rf.1,newdata=effect2.df,type="prob")[,2]

val.rf <- val.prob(pred.valid.rf,effect2.df$Y,pl=F)

roc.valid.rf <- val.rf["C (ROC)"]

r2.valid.rf <- val.rf["R2"]

brier.valid.rf <- val.rf["Brier"]

ici.valid.rf <- val.rf["Eavg"]

E90.valid.rf <- val.rf["E90"]

Emax.valid.rf <- val.rf["Emax"]

brier.max.valid.rf <- 1 - (brier.valid.rf/brier.max.valid)

intercept.valid.rf <- val.rf["Intercept"]

slope.valid.rf <- val.rf["Slope"]

pred.valid.rf <- ifelse(pred.valid.rf==0,0.00001,pred.valid.rf)

negLL <- -sum((effect2.df$Y)\*log(pred.valid.rf) +

(1-effect2.df$Y)\*log(1-pred.valid.rf))

cat(iter,roc.valid.rf,r2.valid.rf,brier.valid.rf,brier.max.valid.rf,

ici.valid.rf,E90.valid.rf,Emax.valid.rf,intercept.valid.rf,slope.valid.rf,

negLL,

file="dgp.rf.rf.out",fill=T,append=T)

remove(rf.1,pred.valid.rf,val.rf,negLL)

################################################################################

# Apply bagged classification trees to the simulated data.

################################################################################

bagg.1 <- randomForest(as.factor(Y) ~ age + female + cshock + acpulmed + sysbp +

diasbp + hrtrate + resp + diabetes + highbp + smokhx + dyslip + famhxcad +

cvatia + angina + cancer + dementia + pud + prevmi + asthma + depres +

perartdis + prevrevasc + chf + hyprthyr + as + hgb + wbc + sod + pot +

glucose + urea + cr,

mtry=33,ntree=n.tree.bagg,

nodesize=tune.bagg$bagg.nodesize,

replace=T,importance=T,

data=effect1.df)

pred.valid.bagg <- predict(bagg.1,newdata=effect2.df,type="prob")[,2]

val.bagg <- val.prob(pred.valid.bagg,effect2.df$Y,pl=F)

roc.valid.bagg <- val.bagg["C (ROC)"]

r2.valid.bagg <- val.bagg["R2"]

brier.valid.bagg <- val.bagg["Brier"]

ici.valid.bagg <- val.bagg["Eavg"]

E90.valid.bagg <- val.bagg["E90"]

Emax.valid.bagg <- val.bagg["Emax"]

brier.max.valid.bagg <- 1 - (brier.valid.bagg/brier.max.valid)

intercept.valid.bagg <- val.bagg["Intercept"]

slope.valid.bagg <- val.bagg["Slope"]

pred.valid.bagg <- ifelse(pred.valid.bagg==0,0.00001,pred.valid.bagg)

negLL <- -sum((effect2.df$Y)\*log(pred.valid.bagg) +

(1-effect2.df$Y)\*log(1-pred.valid.bagg))

cat(iter,roc.valid.bagg,r2.valid.bagg,brier.valid.bagg,brier.max.valid.bagg,

ici.valid.bagg,E90.valid.bagg,Emax.valid.bagg,intercept.valid.bagg,

slope.valid.bagg,negLL,

file="dgp.rf.bagg.out",fill=T,append=T)

remove(bagg.1,pred.valid.bagg,val.bagg,negLL)

################################################################################

# Apply GBM to the simulated data.

################################################################################

gbm.1 <- gbm(Y ~ age + female + cshock + acpulmed + sysbp +

diasbp + hrtrate + resp + diabetes + highbp + smokhx + dyslip + famhxcad +

cvatia + angina + cancer + dementia + pud + prevmi + asthma + depres +

perartdis + prevrevasc + chf + hyprthyr + as + hgb + wbc + sod + pot +

glucose + urea + cr,

data = effect1.df,

distribution = "bernoulli",

n.trees = n.tree.gbm,

interaction.depth = tune.gbm$gbm.interaction.depth,

shrinkage = tune.gbm$gbm.shrinkage,

bag.fraction = 0.5,

train.fraction = 1.0,

cv.folds = 0,

keep.data = TRUE)

pred.valid.gbm <- predict(gbm.1,newdata=effect2.df,n.trees=n.tree.gbm,

type="response")

val.gbm <- val.prob(pred.valid.gbm,effect2.df$Y,pl=F)

roc.valid.gbm <- val.gbm["C (ROC)"]

r2.valid.gbm <- val.gbm["R2"]

brier.valid.gbm <- val.gbm["Brier"]

ici.valid.gbm <- val.gbm["Eavg"]

E90.valid.gbm <- val.gbm["E90"]

Emax.valid.gbm <- val.gbm["Emax"]

brier.max.valid.gbm <- 1 - (brier.valid.gbm/brier.max.valid)

intercept.valid.gbm <- val.gbm["Intercept"]

slope.valid.gbm <- val.gbm["Slope"]

pred.valid.gbm <- ifelse(pred.valid.gbm==0,0.00001,pred.valid.gbm)

negLL <- -sum((effect2.df$Y)\*log(pred.valid.gbm) +

(1-effect2.df$Y)\*log(1-pred.valid.gbm))

cat(iter,roc.valid.gbm,r2.valid.gbm,brier.valid.gbm,brier.max.valid.gbm,

ici.valid.gbm,E90.valid.gbm,Emax.valid.gbm,intercept.valid.gbm,

slope.valid.gbm,negLL,

file="dgp.rf.gbm.out",fill=T,append=T)

remove(gbm.1,pred.valid.gbm,val.gbm,negLL)

################################################################################

# Apply Lasso to the simulated data.

################################################################################

lasso.1 <- glmnet(X.derive,effect1.df$Y,alpha=1,family="binomial",

lambda=tune.lasso$lasso.lambda)

pred.valid.lasso <- predict(lasso.1,X.valid,s=tune.lasso$lasso.lambda,

type="response")

val.lasso <- val.prob(pred.valid.lasso,effect2.df$Y,pl=F)

roc.valid.lasso <- val.lasso["C (ROC)"]

r2.valid.lasso <- val.lasso["R2"]

brier.valid.lasso <- val.lasso["Brier"]

ici.valid.lasso <- val.lasso["Eavg"]

E90.valid.lasso <- val.lasso["E90"]

Emax.valid.lasso <- val.lasso["Emax"]

brier.max.valid.lasso <- 1 - (brier.valid.lasso/brier.max.valid)

intercept.valid.lasso <- val.lasso["Intercept"]

slope.valid.lasso <- val.lasso["Slope"]

pred.valid.lasso <- ifelse(pred.valid.lasso==0,0.00001,pred.valid.lasso)

negLL <- -sum((effect2.df$Y)\*log(pred.valid.lasso) +

(1-effect2.df$Y)\*log(1-pred.valid.lasso))

cat(iter,roc.valid.lasso,r2.valid.lasso,brier.valid.lasso,brier.max.valid.lasso,

ici.valid.lasso,E90.valid.lasso,Emax.valid.lasso,intercept.valid.lasso,

slope.valid.lasso,negLL,

file="dgp.rf.lasso.out",fill=T,append=T)

remove(lasso.1,pred.valid.lasso,val.lasso,negLL)

################################################################################

# Apply ridge regression to the simulated data.

################################################################################

ridge.1 <- glmnet(X.derive,effect1.df$Y,alpha=0,family="binomial",

lambda=tune.ridge$ridge.lambda)

pred.valid.ridge <- predict(ridge.1,X.valid,s=tune.ridge$ridge.lambda,

type="response")

val.ridge <- val.prob(pred.valid.ridge,effect2.df$Y,pl=F)

roc.valid.ridge <- val.ridge["C (ROC)"]

r2.valid.ridge <- val.ridge["R2"]

brier.valid.ridge <- val.ridge["Brier"]

ici.valid.ridge <- val.ridge["Eavg"]

E90.valid.ridge <- val.ridge["E90"]

Emax.valid.ridge <- val.ridge["Emax"]

brier.max.valid.ridge <- 1 - (brier.valid.ridge/brier.max.valid)

intercept.valid.ridge <- val.ridge["Intercept"]

slope.valid.ridge <- val.ridge["Slope"]

pred.valid.ridge <- ifelse(pred.valid.ridge==0,0.00001,pred.valid.ridge)

negLL <- -sum((effect2.df$Y)\*log(pred.valid.ridge) +

(1-effect2.df$Y)\*log(1-pred.valid.ridge))

cat(iter,roc.valid.ridge,r2.valid.ridge,brier.valid.ridge,brier.max.valid.ridge,

ici.valid.ridge,E90.valid.ridge,Emax.valid.ridge,intercept.valid.ridge,

slope.valid.ridge,negLL,

file="dgp.rf.ridge.out",fill=T,append=T)

remove(ridge.1,pred.valid.ridge,val.ridge,negLL)

}