Torbet McNeil, Assignment 3

2.

library("caret")

library("knitr")

library("mlbench")

library("parallel")

library("doParallel")

library("foreach")

library("haven")

library("MASS")

library("ggplot2")

library("randomForest")

library("pROC")

library("party")

library("dplyr")

library("ggraph")

library("igraph")

library("rpart.plot")

#

cl <- makeCluster(detectCores() - 1)

CCES2016 <- read\_dta("C:/Users/um181144/Downloads/CCES2016\_abbreviated.dta")

#

dat <- na.omit(CCES2016)

dat$voted2016 <- factor(dat$voted2016, labels=c("No\_Vote", "Voted"))

#

#Split data

set.seed(1985)

trainIndex <- createDataPartition(dat$voted2016, p=0.2, list = FALSE, times = 1)

#

train <- dat[trainIndex,]

test <- dat[-trainIndex,]

#Set control parameters for model training

#

fitCtrl <- trainControl(method = "repeatedcv",

number = 5,

repeats = 2,

summaryFunction=twoClassSummary,

## Estimate class probabilities

classProbs = TRUE,

## returnData = TRUE,

savePredictions = TRUE,

## Search "grid" or "random"

search = "random",

## Down-sampling

sampling = "down",

## Use cluster

allowParallel = TRUE)

# Fit the classification tree

#

singletree <- rpart(voted2016 ~ ., data = train,

control = rpart.control(minsplit = 20, minbucket = 5))

#

singletree$variable.importance

#

# age voted2016primary education

# 330.41610 212.24077 14.02922

# partyid ideology income

# 13.57184 12.69997 10.96703

#

#For a single decision tree, the training accuracy was 65.29%, and the testing accuracy 65.43%. #The three most important features from the single decision tree were age, voted in the 2016 #primary, and education level.

#

# Plot tree

#

rpart.plot(singletree)

#

train.pred <- predict(singletree, train, type="class")

test.pred <- predict(singletree, test, type="class")

#

install.packages("e1071")

library("e1071")

confusionMatrix(train$voted2016, train.pred)

#

#Confusion Matrix and Statistics

Reference

Prediction No\_Vote Voted

No\_Vote 3859 1267

Voted 2141 2551

Accuracy : 0.6529

95% CI : (0.6434, 0.6623)

No Information Rate : 0.6111

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.2989

Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.6432

Specificity : 0.6682

Pos Pred Value : 0.7528

Neg Pred Value : 0.5437

Prevalence : 0.6111

Detection Rate : 0.3931

Detection Prevalence : 0.5221

Balanced Accuracy : 0.6557

'Positive' Class : No\_Vote

#

confusionMatrix(test$voted2016, test.pred)

#

Confusion Matrix and Statistics

#

# Reference

Prediction No\_Vote Voted

No\_Vote 15482 5021

Voted 8555 10211

Accuracy : 0.6543

95% CI : (0.6496, 0.659)

No Information Rate : 0.6121

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.3016

Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.6441

Specificity : 0.6704

Pos Pred Value : 0.7551

Neg Pred Value : 0.5441

Prevalence : 0.6121

Detection Rate : 0.3943

Detection Prevalence : 0.5221

Balanced Accuracy : 0.6572

'Positive' Class : No\_Vote

#

# Set testing grid for random forest

#

rfGrid <- expand.grid(mtry = 1:6)

#

set.seed(1985)

rf.res <- train(voted2016 ~ .,

data=train,

method="rf",

trControl=fitCtrl,

tuneGrid=rfGrid,

#tuneLength=10,

metric="ROC",

verbose=FALSE)

#

rf.res

#

#Random Forest

#

9818 samples

9 predictor

2 classes: 'No\_Vote', 'Voted'

No pre-processing

Resampling: Cross-Validated (5 fold, repeated 2 times)

Summary of sample sizes: 7854, 7854, 7855, 7854, 7855, 7855, ...

Addtional sampling using down-sampling

Resampling results across tuning parameters:

mtry ROC Sens Spec

1 0.7158233 0.6091458 0.7072659

2 0.7185182 0.6764489 0.6592073

3 0.7007165 0.6384087 0.6622986

4 0.6861366 0.6285583 0.6455660

5 0.6819077 0.6241694 0.6445006

6 0.6798796 0.6247541 0.6435439

ROC was used to select the optimal model using the largest value.

The final value used for the model was mtry = 2.

#

#

# Extract predictions

#

confusionMatrix(predict(rf.res, train, type="raw"), train$voted2016)

#

#Confusion Matrix and Statistics

#

Reference

Prediction No\_Vote Voted

No\_Vote 4019 892

Voted 1107 3800

Accuracy : 0.7964

95% CI : (0.7883, 0.8043)

No Information Rate : 0.5221

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.5928

Mcnemar's Test P-Value : 1.698e-06

Sensitivity : 0.7840

Specificity : 0.8099

Pos Pred Value : 0.8184

Neg Pred Value : 0.7744

Prevalence : 0.5221

Detection Rate : 0.4094

Detection Prevalence : 0.5002

Balanced Accuracy : 0.7970

'Positive' Class : No\_Vote

#

confusionMatrix(predict(rf.res, test, type="raw"), test$voted2016)

#

#Confusion Matrix and Statistics

#

Reference

Prediction No\_Vote Voted

No\_Vote 13883 6449

Voted 6620 12317

Accuracy : 0.6672

95% CI : (0.6625, 0.6719)

No Information Rate : 0.5221

P-Value [Acc > NIR] : <2e-16

Kappa : 0.3333

Mcnemar's Test P-Value : 0.137

Sensitivity : 0.6771

Specificity : 0.6563

Pos Pred Value : 0.6828

Neg Pred Value : 0.6504

Prevalence : 0.5221

Detection Rate : 0.3535

Detection Prevalence : 0.5178

Balanced Accuracy : 0.6667

'Positive' Class : No\_Vote

#

# Variable importance

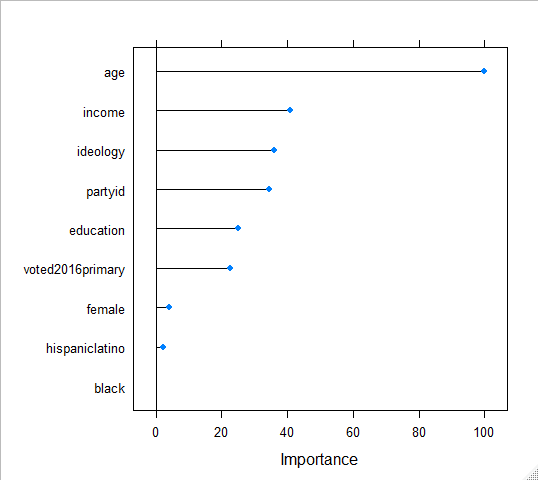
#

rfImp <- varImp(rf.res)

plot(rfImp)

#

##For the random forest, the training accuracy was 79.64%, and the testing accuracy was 66.72%. The three most important features from the random forest were age, income, and ideology.



#gbm

install.packages("gbm")

library("gbm")

install.packages("qadprog")

install.packages("iml")

install.packages("nnet")

library("quadprog")

library("iml")

library("nnet")

#

# Set control parameters for model training

#

fitCtrl <- trainControl(method = "repeatedcv",

number = 5,

repeats = 2,

## returnData = TRUE,

savePredictions = TRUE,

## Search "grid" or "random"

search = "random",

#sampling = "down",

## Use cluster

allowParallel = TRUE)

#

gbmGrid <- expand.grid(n.trees = c(1:20)\*100,

interaction.depth=c(2,3),

shrinkage = c(0.01, 0.05),

n.minobsinnode=c(5,10))

#

#Estimate GBM model

set.seed(1985)

gbm.res <- train(voted2016 ~ .,

data=train,

method="gbm",

trControl=fitCtrl,

tuneGrid=gbmGrid,

metric="Accuracy",

verbose=FALSE)

#

gbm.res

#

#Stochastic Gradient Boosting

#

9818 samples

9 predictor

2 classes: 'No\_Vote', 'Voted'

No pre-processing

Resampling: Cross-Validated (5 fold, repeated 2 times)

Summary of sample sizes: 7854, 7854, 7855, 7854, 7855, 7855, ...

Resampling results across tuning parameters:

shrinkage interaction.depth n.minobsinnode n.trees Accuracy Kappa

0.01 2 5 100 0.6483494 0.2868512

0.01 2 5 200 0.6574146 0.3073975

0.01 2 5 300 0.6627624 0.3191103

0.01 2 5 400 0.6624060 0.3186569

0.01 2 5 500 0.6629662 0.3200214

0.01 2 5 600 0.6643415 0.3231133

0.01 2 5 700 0.6646980 0.3241570

0.01 2 5 800 0.6669894 0.3291194

0.01 2 5 900 0.6679570 0.3313658

0.01 2 5 1000 0.6682117 0.3320412

0.01 2 5 1100 0.6688229 0.3333606

0.01 2 5 1200 0.6695357 0.3348564

0.01 2 5 1300 0.6690263 0.3338404

0.01 2 5 1400 0.6689245 0.3336741

0.01 2 5 1500 0.6688735 0.3335881

0.01 2 5 1600 0.6689754 0.3338624

0.01 2 5 1700 0.6697392 0.3354735

0.01 2 5 1800 0.6698411 0.3356837

0.01 2 5 1900 0.6698409 0.3357518

0.01 2 5 2000 0.6700957 0.3363760

0.01 2 10 100 0.6492152 0.2885075

0.01 2 10 200 0.6579240 0.3084897

0.01 2 10 300 0.6622022 0.3179541

0.01 2 10 400 0.6625078 0.3189423

0.01 2 10 500 0.6634246 0.3209794

0.01 2 10 600 0.6640871 0.3225698

0.01 2 10 700 0.6647488 0.3242799

0.01 2 10 800 0.6676007 0.3303198

0.01 2 10 900 0.6679060 0.3312462

0.01 2 10 1000 0.6674987 0.3305891

0.01 2 10 1100 0.6682117 0.3321171

0.01 2 10 1200 0.6689247 0.3336175

0.01 2 10 1300 0.6697394 0.3353246

0.01 2 10 1400 0.6691790 0.3342950

0.01 2 10 1500 0.6691282 0.3342154

0.01 2 10 1600 0.6691789 0.3343877

0.01 2 10 1700 0.6696883 0.3354326

0.01 2 10 1800 0.6694844 0.3350173

0.01 2 10 1900 0.6696373 0.3353475

0.01 2 10 2000 0.6701975 0.3365335

0.01 3 5 100 0.6533915 0.2978210

0.01 3 5 200 0.6614384 0.3162475

0.01 3 5 300 0.6632207 0.3203957

0.01 3 5 400 0.6650541 0.3247173

0.01 3 5 500 0.6671929 0.3296571

0.01 3 5 600 0.6696376 0.3349956

0.01 3 5 700 0.6694848 0.3348807

0.01 3 5 800 0.6704013 0.3369162

0.01 3 5 900 0.6710633 0.3382854

0.01 3 5 1000 0.6716743 0.3395445

0.01 3 5 1100 0.6707576 0.3377805

0.01 3 5 1200 0.6713175 0.3389381

0.01 3 5 1300 0.6716229 0.3396278

0.01 3 5 1400 0.6723870 0.3412877

0.01 3 5 1500 0.6727946 0.3421596

0.01 3 5 1600 0.6722853 0.3412143

0.01 3 5 1700 0.6719797 0.3406932

0.01 3 5 1800 0.6723361 0.3414767

0.01 3 5 1900 0.6727944 0.3424553

0.01 3 5 2000 0.6724889 0.3419417

0.01 3 10 100 0.6522709 0.2953899

0.01 3 10 200 0.6605727 0.3144409

0.01 3 10 300 0.6638318 0.3216109

0.01 3 10 400 0.6652581 0.3250593

0.01 3 10 500 0.6681101 0.3315196

0.01 3 10 600 0.6698414 0.3354396

0.01 3 10 700 0.6711653 0.3382856

0.01 3 10 800 0.6708598 0.3377630

0.01 3 10 900 0.6715216 0.3392457

0.01 3 10 1000 0.6717252 0.3397284

0.01 3 10 1100 0.6719289 0.3401852

0.01 3 10 1200 0.6716232 0.3396129

0.01 3 10 1300 0.6710631 0.3384903

0.01 3 10 1400 0.6721325 0.3407162

0.01 3 10 1500 0.6724380 0.3414155

0.01 3 10 1600 0.6716741 0.3399542

0.01 3 10 1700 0.6717252 0.3401523

0.01 3 10 1800 0.6717251 0.3402606

0.01 3 10 1900 0.6710631 0.3389345

0.01 3 10 2000 0.6713686 0.3396446

0.05 2 5 100 0.6625080 0.3190855

0.05 2 5 200 0.6678044 0.3312667

0.05 2 5 300 0.6689248 0.3337068

0.05 2 5 400 0.6701977 0.3365826

0.05 2 5 500 0.6702482 0.3369406

0.05 2 5 600 0.6696880 0.3359572

0.05 2 5 700 0.6697389 0.3362798

0.05 2 5 800 0.6694843 0.3358807

0.05 2 5 900 0.6690767 0.3352177

0.05 2 5 1000 0.6694333 0.3360700

0.05 2 5 1100 0.6695353 0.3363162

0.05 2 5 1200 0.6676001 0.3324888

0.05 2 5 1300 0.6674983 0.3323156

0.05 2 5 1400 0.6661741 0.3298240

0.05 2 5 1500 0.6664796 0.3304833

0.05 2 5 1600 0.6653593 0.3282820

0.05 2 5 1700 0.6661739 0.3300008

0.05 2 5 1800 0.6651044 0.3279308

0.05 2 5 1900 0.6649011 0.3275583

0.05 2 5 2000 0.6651556 0.3281278

0.05 2 10 100 0.6609289 0.3160239

0.05 2 10 200 0.6679059 0.3314642

0.05 2 10 300 0.6682621 0.3324229

0.05 2 10 400 0.6688733 0.3340345

0.05 2 10 500 0.6707572 0.3379615

0.05 2 10 600 0.6692804 0.3352901

0.05 2 10 700 0.6697387 0.3363525

0.05 2 10 800 0.6696876 0.3363535

0.05 2 10 900 0.6693313 0.3358067

0.05 2 10 1000 0.6692803 0.3358538

0.05 2 10 1100 0.6682108 0.3338442

0.05 2 10 1200 0.6680582 0.3335557

0.05 2 10 1300 0.6673963 0.3322761

0.05 2 10 1400 0.6671926 0.3318785

0.05 2 10 1500 0.6666324 0.3309236

0.05 2 10 1600 0.6664286 0.3305270

0.05 2 10 1700 0.6674980 0.3327130

0.05 2 10 1800 0.6664288 0.3304897

0.05 2 10 1900 0.6660721 0.3299182

0.05 2 10 2000 0.6662758 0.3302589

0.05 3 5 100 0.6667855 0.3289144

0.05 3 5 200 0.6706561 0.3375034

0.05 3 5 300 0.6710633 0.3386910

0.05 3 5 400 0.6715215 0.3400060

0.05 3 5 500 0.6719287 0.3411211

0.05 3 5 600 0.6697386 0.3368620

0.05 3 5 700 0.6690769 0.3357215

0.05 3 5 800 0.6683125 0.3343242

0.05 3 5 900 0.6682616 0.3343833

0.05 3 5 1000 0.6683633 0.3345828

0.05 3 5 1100 0.6674466 0.3328400

0.05 3 5 1200 0.6661735 0.3304307

0.05 3 5 1300 0.6647983 0.3276781

0.05 3 5 1400 0.6649513 0.3280539

0.05 3 5 1500 0.6651548 0.3285100

0.05 3 5 1600 0.6655113 0.3292716

0.05 3 5 1700 0.6644930 0.3272529

0.05 3 5 1800 0.6642893 0.3269684

0.05 3 5 1900 0.6630159 0.3242733

0.05 3 5 2000 0.6632197 0.3246867

0.05 3 10 100 0.6670913 0.3296848

0.05 3 10 200 0.6717257 0.3396812

0.05 3 10 300 0.6708086 0.3381815

0.05 3 10 400 0.6722341 0.3415086

0.05 3 10 500 0.6710631 0.3394082

0.05 3 10 600 0.6706556 0.3388426

0.05 3 10 700 0.6701972 0.3381053

0.05 3 10 800 0.6688730 0.3355643

0.05 3 10 900 0.6682618 0.3343286

0.05 3 10 1000 0.6670906 0.3320849

0.05 3 10 1100 0.6678543 0.3335978

0.05 3 10 1200 0.6669887 0.3319845

0.05 3 10 1300 0.6678542 0.3337842

0.05 3 10 1400 0.6663265 0.3307710

0.05 3 10 1500 0.6652058 0.3284876

0.05 3 10 1600 0.6646964 0.3275500

0.05 3 10 1700 0.6642385 0.3266104

0.05 3 10 1800 0.6622014 0.3225077

0.05 3 10 1900 0.6620485 0.3221690

0.05 3 10 2000 0.6622013 0.3225517

Accuracy was used to select the optimal model using the largest value.

The final values used for the model were n.trees = 1500, interaction.depth =

3, shrinkage = 0.01 and n.minobsinnode = 5.

#

#

# Extract predictions (gbm)

confusionMatrix(predict(gbm.res, train, type="raw"), train$voted2016)

#

#Confusion Matrix and Statistics

#

Reference

Prediction No\_Vote Voted

No\_Vote 3749 1736

Voted 1377 2956

Accuracy : 0.6829

95% CI : (0.6736, 0.6921)

No Information Rate : 0.5221

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.3626

Mcnemar's Test P-Value : 1.395e-10

Sensitivity : 0.7314

Specificity : 0.6300

Pos Pred Value : 0.6835

Neg Pred Value : 0.6822

Prevalence : 0.5221

Detection Rate : 0.3818

Detection Prevalence : 0.5587

Balanced Accuracy : 0.6807

'Positive' Class : No\_Vote

#

confusionMatrix(predict(gbm.res, test, type="raw"), test$voted2016)

#

#Confusion Matrix and Statistics

#

Reference

Prediction No\_Vote Voted

No\_Vote 14757 7144

Voted 5746 11622

Accuracy : 0.6718

95% CI : (0.6671, 0.6764)

No Information Rate : 0.5221

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.3401

Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.7197

Specificity : 0.6193

Pos Pred Value : 0.6738

Neg Pred Value : 0.6692

Prevalence : 0.5221

Detection Rate : 0.3758

Detection Prevalence : 0.5577

Balanced Accuracy : 0.6695

'Positive' Class : No\_Vote

#

# IML: Interpretable Machine Learning

#

X <- dat[which(names(dat) != "voted2016")]

predictor <- Predictor$new(gbm.res, data = X, y = dat$voted2016)

#

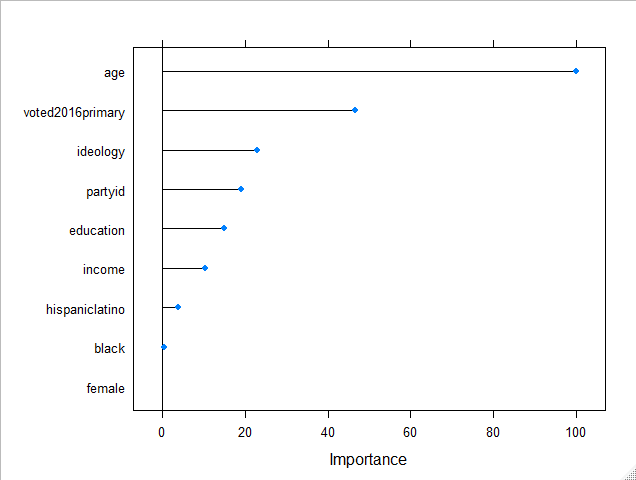
# Measure feature importance (permutation test)

gbmImp <- varImp(gbm.res)

plot(gbmImp)

#

#For the GBM, the training accuracy was 68.29%, and the testing accuracy was 67.18%. The #three most important features from the GBM were age, voted in 2016 primary, and ideology.



3.

# Use Shapley values to measure feature effects

#

X <- dat[which(names(dat) != "voted2016")]

predictor <- Predictor$new(rf.res, data = X, y = dat$voted2016)

#

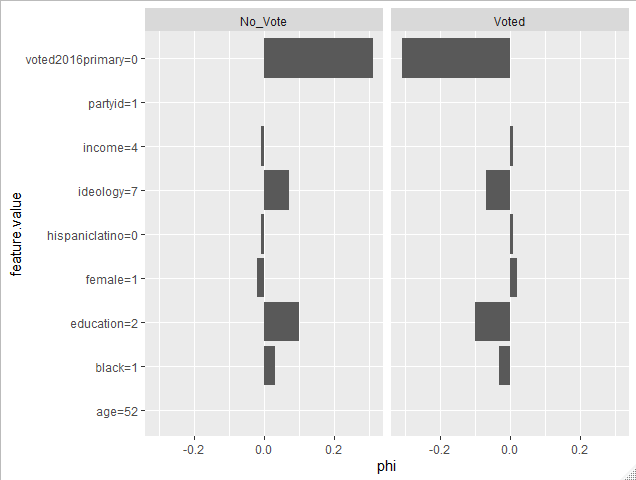
shapley <- Shapley$new(predictor, x.interest = X[1,])

shapley$plot()

shapley$results

#

#I used Shapley values to predict the effects of age on voting turnout from the random forest. In #this observation, age does not influence the predicted value of the model as the Shapley value is #0.00; age had the 8th/9th smallest contribution in magnitude to the model.



feature class phi phi.var feature.value

1 age No\_Vote 0.00 0.12121212 age=52

2 female No\_Vote -0.02 0.01979798 female=1

3 education No\_Vote 0.10 0.09090909 education=2

4 black No\_Vote 0.03 0.04959596 black=1

5 hispaniclatino No\_Vote -0.01 0.01000000 hispaniclatino=0

6 voted2016primary No\_Vote 0.31 0.21606061 voted2016primary=0

7 ideology No\_Vote 0.07 0.08595960 ideology=7

8 partyid No\_Vote 0.00 0.04040404 partyid=1

9 income No\_Vote -0.01 0.07060606 income=4

10 age Voted 0.00 0.12121212 age=52

11 female Voted 0.02 0.01979798 female=1

12 education Voted -0.10 0.09090909 education=2

13 black Voted -0.03 0.04959596 black=1

14 hispaniclatino Voted 0.01 0.01000000 hispaniclatino=0

15 voted2016primary Voted -0.31 0.21606061 voted2016primary=0

16 ideology Voted -0.07 0.08595960 ideology=7

17 partyid Voted 0.00 0.04040404 partyid=1

18 income Voted 0.01 0.07060606 income=4