Practical Machine Learning Course Project

Jim Rhudy

12/21/2021

## Question

This project is concerned with prediction of correct versus incorrect performance of barbell lifts (total of five classes) by four individuals contributing a variety of accelerometric features in three axes from body-worn sensors to a dataset described here (source:<http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har>).

## Data

#Preparation of data and required packages  
trainDat <- read.csv("C:/Users/Jim/Desktop/Rwork/jhuDataScienceFund/machine/project/pml-training.csv", header=TRUE, sep=",")  
dim(trainDat) #[1] 19622 160

## [1] 19622 160

testDat <- read.csv("C:/Users/Jim/Desktop/Rwork/jhuDataScienceFund/machine/project/pml-testing.csv", header=TRUE, sep=",")  
  
library(tidyverse); library(caret)

## Feature selection from from training set

############### remove all non-accelerometric features  
accel <- trainDat[ , -c(1:7)]  
dim(accel)

## [1] 19622 153

################### remove features with >90% missing  
accel <- accel[, which(colMeans(!is.na(accel)) > 0.9)] #%>% glimpse  
dim(accel)

## [1] 19622 86

################### remove near zero variance features  
nsv <- nearZeroVar(accel)  
accel <- accel[, -nsv]  
dim(accel)

## [1] 19622 53

################ split into training and validation  
set.seed(2021)  
inTrain <- createDataPartition(y=accel$classe, p=0.75, list=FALSE)  
training <- accel[inTrain,]  
dim(training) #[1] 14718 53

## [1] 14718 53

validation <- accel[-inTrain,]  
dim(validation) #[1] 4904 53

## [1] 4904 53

###################### inspect distribution of classe in training set  
dat <- training %>%  
 group\_by(classe) %>%  
 summarise(count=n()) %>% glimpse

## Rows: 5  
## Columns: 2  
## $ classe <chr> "A", "B", "C", "D", "E"  
## $ count <int> 4185, 2848, 2567, 2412, 2706

## Algorithms

set.seed(2021)  
  
############################### source for implementation of parallel processing:  
#https://github.com/lgreski/datasciencectacontent/blob/master/markdown/pml-randomForestPerformance.md  
  
# The Process: A Parallel Implementation of Random Forest  
  
#Step 1: Configure parallel processing  
library(parallel); library(doParallel)  
cluster <- makeCluster(detectCores() - 1)   
registerDoParallel(cluster)  
  
#Step 2: Configure trainControl object  
fitControl <- trainControl(method = "cv",  
 number = 5,  
 allowParallel = TRUE)  
  
#Step 3: Develop training model  
y <- training[,53]  
length(y) #14718

## [1] 14718

x <- training[,-53]  
dim(x) #14718 52

## [1] 14718 52

system.time(fit <- train(x,y, method="rf",data=training,trControl = fitControl))

## user system elapsed   
## 40.93 0.51 461.51

#Step 4: De-register parallel processing cluster  
stopCluster(cluster)  
registerDoSEQ()  
  
########################## parallel processing guidance ends  
  
#assess properties of the model  
fit

## Random Forest   
##   
## 14718 samples  
## 52 predictor  
## 5 classes: 'A', 'B', 'C', 'D', 'E'   
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold)   
## Summary of sample sizes: 11775, 11774, 11775, 11775, 11773   
## Resampling results across tuning parameters:  
##   
## mtry Accuracy Kappa   
## 2 0.9919147 0.9897716  
## 27 0.9915069 0.9892561  
## 52 0.9833540 0.9789436  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was mtry = 2.

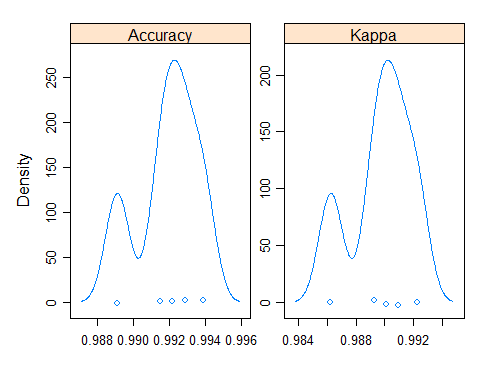
fit$resample

## Accuracy Kappa Resample  
## 1 0.9938838 0.9922638 Fold1  
## 2 0.9921875 0.9901164 Fold2  
## 3 0.9915110 0.9892622 Fold5  
## 4 0.9891267 0.9862435 Fold4  
## 5 0.9928644 0.9909723 Fold3

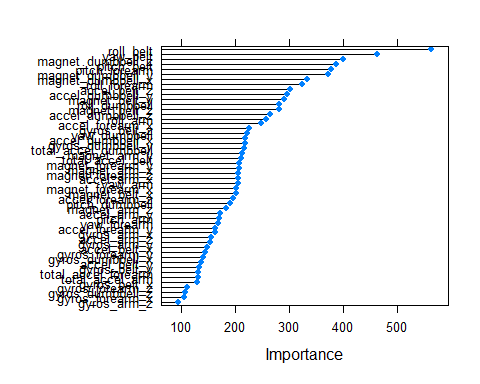
confusionMatrix.train(fit)

## Cross-Validated (5 fold) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction A B C D E  
## A 28.4 0.1 0.0 0.0 0.0  
## B 0.0 19.1 0.2 0.0 0.0  
## C 0.0 0.1 17.2 0.3 0.0  
## D 0.0 0.0 0.0 16.1 0.0  
## E 0.0 0.0 0.0 0.0 18.4  
##   
## Accuracy (average) : 0.9919

#the training model now has an accuracy of 0.9919  
  
# density plot of accuracy and concordance  
resampleHist(fit)



# accuracy of the model peaks at 0.992. concordance of the model peaks at 99.0%  
  
# interpret variable importance  
rfImp <-varImp(fit, scale=FALSE)   
plot(rfImp) # shows features in descending order of importance



# predictions on validation data  
predictions <- predict(fit,newdata=validation)  
  
#confusion matrix  
confusionMatrix(factor(predictions), factor(validation$classe))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1395 4 0 0 0  
## B 0 944 0 0 0  
## C 0 1 854 20 3  
## D 0 0 1 784 3  
## E 0 0 0 0 895  
##   
## Overall Statistics  
##   
## Accuracy : 0.9935   
## 95% CI : (0.9908, 0.9955)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9917   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 1.0000 0.9947 0.9988 0.9751 0.9933  
## Specificity 0.9989 1.0000 0.9941 0.9990 1.0000  
## Pos Pred Value 0.9971 1.0000 0.9727 0.9949 1.0000  
## Neg Pred Value 1.0000 0.9987 0.9998 0.9951 0.9985  
## Prevalence 0.2845 0.1935 0.1743 0.1639 0.1837  
## Detection Rate 0.2845 0.1925 0.1741 0.1599 0.1825  
## Detection Prevalence 0.2853 0.1925 0.1790 0.1607 0.1825  
## Balanced Accuracy 0.9994 0.9974 0.9965 0.9871 0.9967

#the model fit on the validation data now has an accuracy of 0.9939

The trainControl() step prepares for cross validation with 5-fold resampling rather than the default bootstrap. This choice may have resulted in reduced model accuracy as a tradeoff for increased processing performance.

The model fit on the training data has accuracy of 0.9919; the same model fit on the validation data has accuracy of 0.9939.

## In and Out of Sample Error

The off-diagonal elements of the validation confusion matrix sum to 30; this represents an out-of-sample error rate of 0.6% of the total 4904.

## Prediction of Test Cases

Once accuracy greater than 99% was achieved, prediction of test cases was undertaken and the final project quiz was completed.

## Explanation of choices and accuracy tradeoffs

The random forest method versus other methods has the advantage of accuracy along with the disadvantages of relatively slow computation, difficult interpretability, and tendency to overfit. The choice of cross validation with 5-fold resampling rather than the default bootstrap may have resulted in reduced model accuracy as a tradeoff for increased processing performance.