Practical Machine Learning Course Project

Albert Shuxiang Li

January 31, 2016

Project Background

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data Source of Project

The training data for this project are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har.

Project Objective

The goal of this project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. Any of the other variables can be used to predict with. Following questions will be answered in this write up.

1. How the prediction models are built?
2. How cross validation being used?
3. What the expected out of sample error is?
4. Why the choices of prediction model?
5. Show prediction result for 20 different test cases with chosen prediction model?

Process Data

Download Data and Load into System

```r
set.seed(3606)
library(caret); library(rpart); library(rpart.plot); library(RColorBrewer)
library(rattle); library(randomForest); library(knitr); library(gbm)
```

```
Url_train <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
Url_test <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read.csv(url(Url_train), na.strings=c("NA", "#DIV/0!", ""))
testing <- read.csv(url(Url_test), na.strings=c("NA", "#DIV/0!", ""))
dim(training); dim(testing)
```

Clean Data

Remove NearZeroVariance variables

```
nzv <- nearZeroVar(training, saveMetrics = TRUE)
training <- training[, nzv$nzv==FALSE]
```

Remove variables which contain more than 55% NA

```
trainingTEMP <- training
for(i in 1:length(training)) {
    if( sum( is.na( training[, i] ) ) /nrow(training) >= .55) {
        for(j in 1:length(trainingTEMP)) {
            if( length( grep(names(training[i]), names(trainingTEMP)[j]) ) == 1)  {
                trainingTEMP <- trainingTEMP[ , -j]
            }
        }
    }
}
training <- trainingTEMP; rm(trainingTEMP)
# dim(training); dim(testing)
```

Remove irrelavent variables

After check with str(training), it is recongnized that the first 6 columns (variables) are NOT related to movement, thus we can remove:
1. X
2. user_name
3. raw_timestamp_part_1
4. raw_timestamp_part_2
5. cvtd_timestamp
6. num_window

```
training = training[,-c(1:6)]
```

Remove highly correlated variables

```
x1 <- training[, -53]; correlation.matrix <- cor(x1)
highly.correlated <- findCorrelation(correlation.matrix, cutoff=0.75)
# print(names(training)[highly.correlated])
```

```r
training <- training[, -highly.correlated]
# dim(training); str(training)
```

Partition training set into two sets

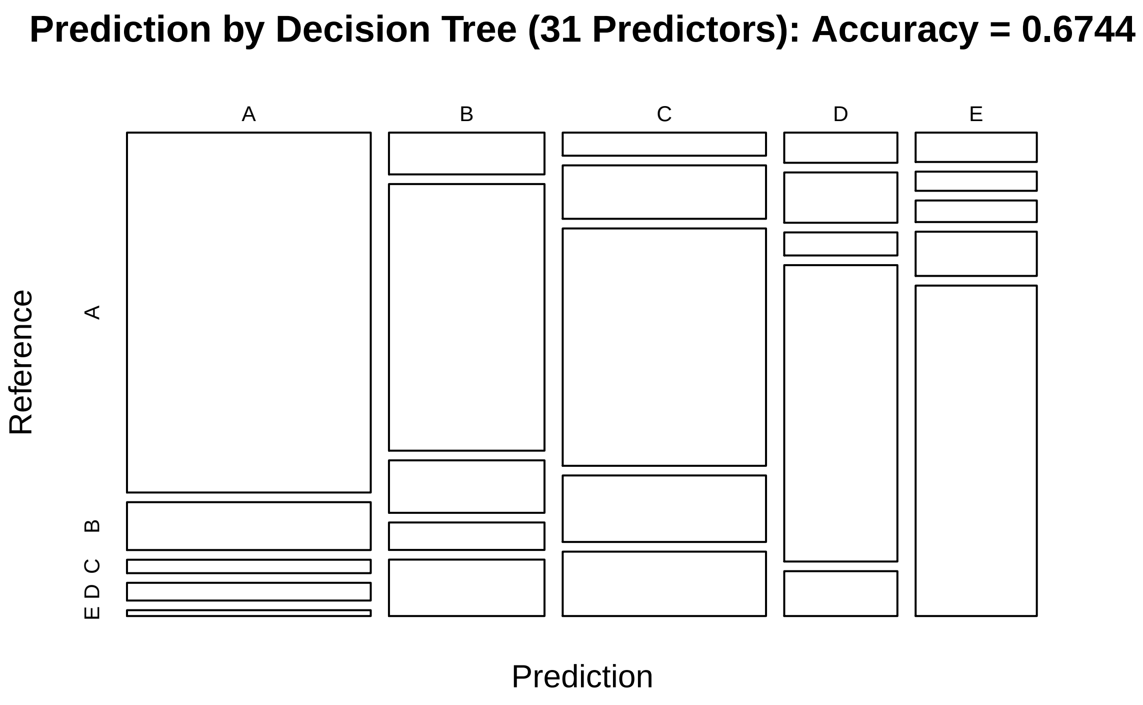
```r
inTrain <- createDataPartition(training$classe, p=0.6, list=FALSE)
myTraining <- training[inTrain, ]
myTesting <- training[-inTrain, ]
# dim(myTraining); dim(myTesting); dim(testing)
```

Sync myTesting, myTesting and testing columns

```r
myTesting <- myTesting[colnames(myTraining)]
testing <- testing[colnames(myTraining[, -32])] # no classe variable in testing
# dim(myTraining); dim(myTesting); dim(testing)
# therefore, there are 31 predictors being used
```

Predict Utilizing Decision Trees

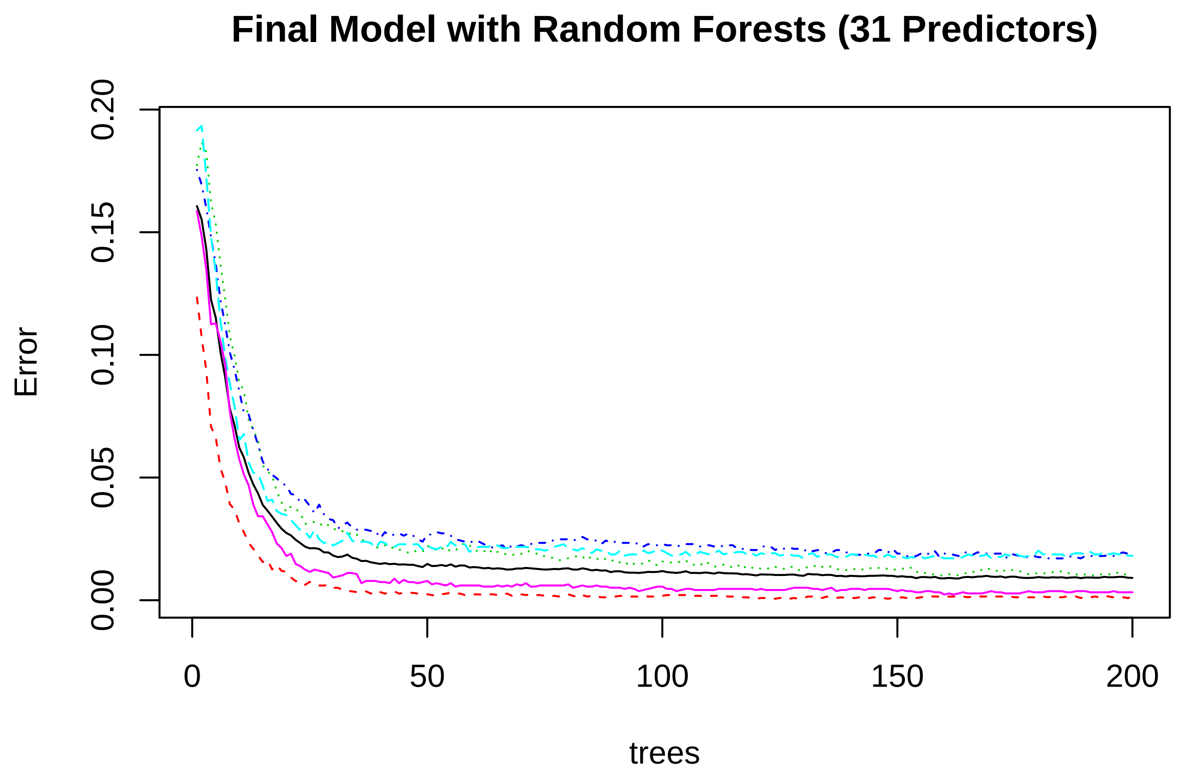
```r
# t1 <- Sys.time()
set.seed(3606)
modFitDT <- rpart::rpart(classe ~ ., data=myTraining, method="class")
# rattle::fancyRpartPlot(modFitDT)
predictionsDT <- predict(modFitDT, myTesting, type = "class")
confusionMatrixDT <- confusionMatrix(predictionsDT, myTesting$classe)
# confusionMatrixDT
plot(confusionMatrixDT$table, col = confusionMatrixDT$byClass,
     main = paste("Prediction by Decision Tree (31 Predictors): Accuracy =",
                  round(confusionMatrixDT$overall['Accuracy'], 4)))
```



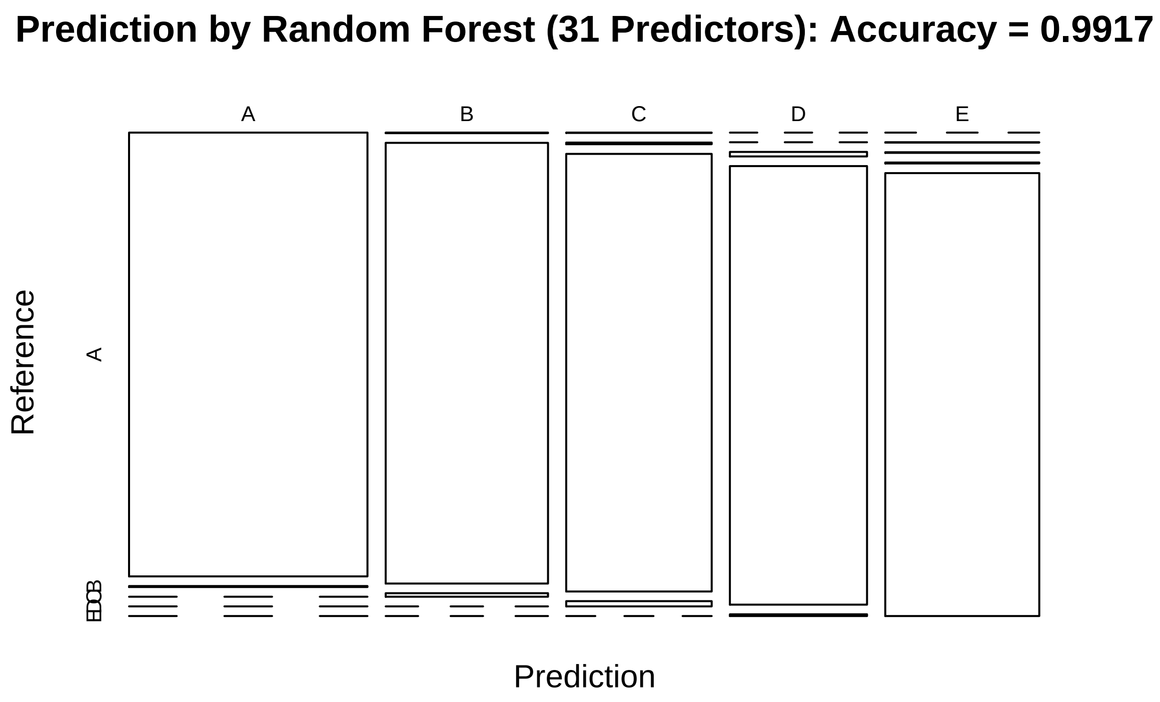
```
# t2 <- Sys.time(); t2-t1 ## Time difference of 1.508441 secs
```

Predict Utilizing Random Forest

```
# t3 <- Sys.time()
set.seed(3606)
modFitRF <- randomForest(classe ~ ., data=myTraining, ntree=200, importance=TRUE)
predictionRF <- predict(modFitRF, myTesting, type = "class")
ConfusionMatrixRF <- confusionMatrix(predictionRF, myTesting$classe)
# ConfusionMatrixRF;
plot(modFitRF, main="Final Model with Random Forests (31 Predictors)")
```



```
plot(ConfusionMatrixRF$table, col = ConfusionMatrixRF$byClass,
     main = paste("Prediction by Random Forest (31 Predictors): Accuracy =",
                  round(ConfusionMatrixRF$overall['Accuracy'], 4)))
```



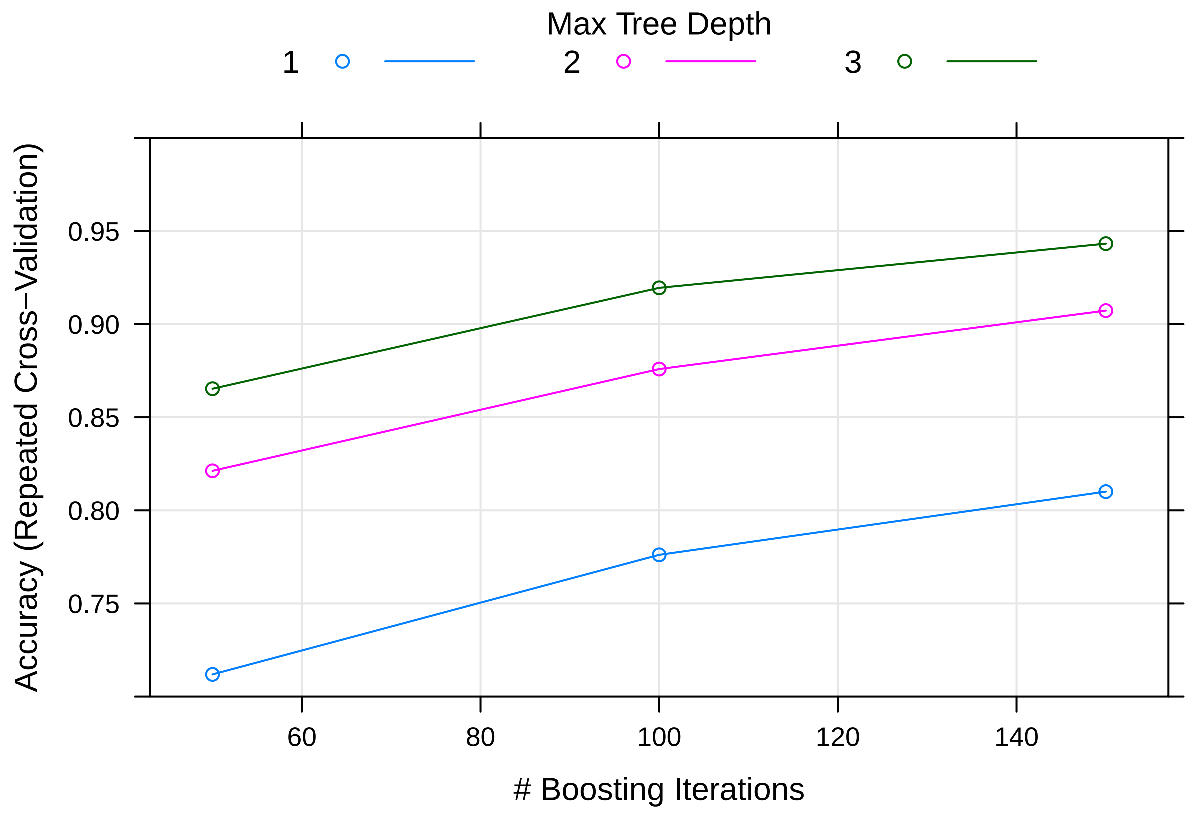
```
# t4 <- Sys.time(); t4 - t3 ## Time difference of 14.26705 secs
```

Predict Utilizing Gradient Boosting Machine

```
# t5 <- Sys.time()
set.seed(3606)
library(doParallel)
cluster <- makeCluster(detectCores() - 1)
registerDoParallel(cluster)
fitControlGBM <- caret::trainControl(method = "repeatedcv",
                                     number = 5, repeats = 4, allowParallel = TRUE)
modFitGBM <- caret::train(classe ~ ., data = myTraining,
                          method = "gbm", trControl = fitControlGBM, verbose = FALSE)
prodictionGBM <- predict(modFitGBM, newdata=myTesting)
stopCluster(cluster)
ConfusionMatrixGBM <- confusionMatrix(prodictionGBM, myTesting$classe)
cat("The GBM Accuracy = ", round(ConfusionMatrixGBM$overall['Accuracy']*100, 2), "%", sep="")
```

```
## The GBM Accuracy = 94.72%
```

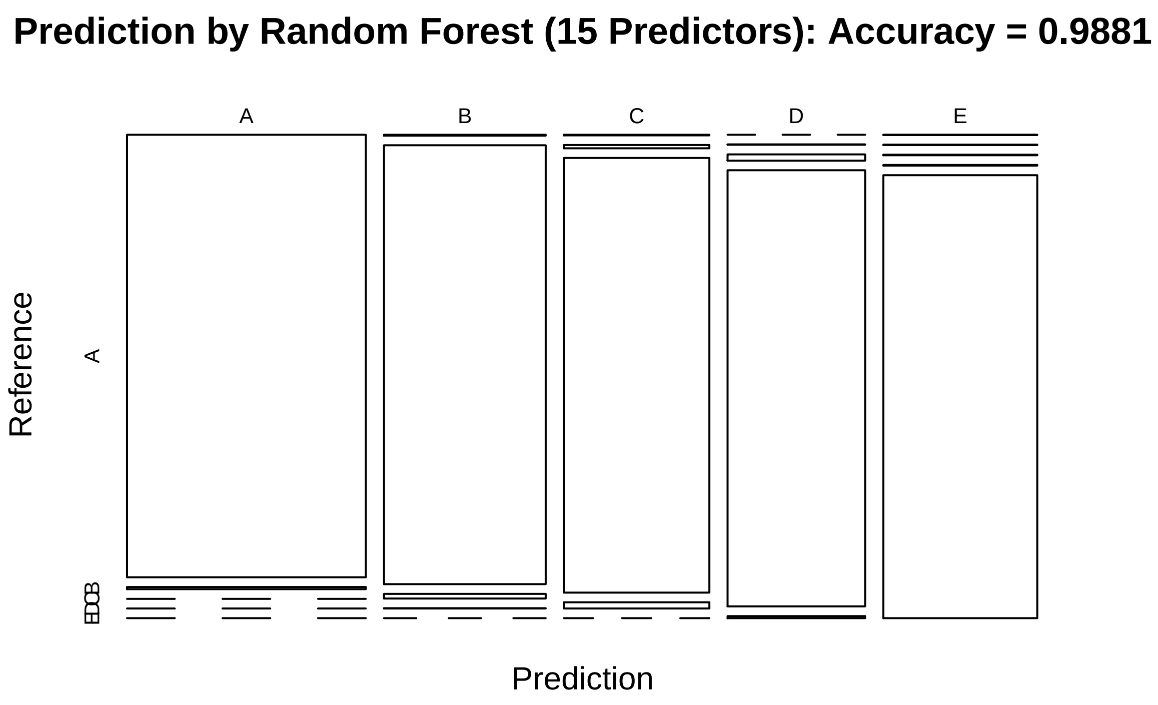
```r
plot(modFitGBM, ylim=c(0.7, 1))
```



```r
# t6 <- Sys.time(); t6 - t5 ## Time difference of 3.397989 mins
```

Predict Utilizing Random Forest plus Recusive Feature Elimination

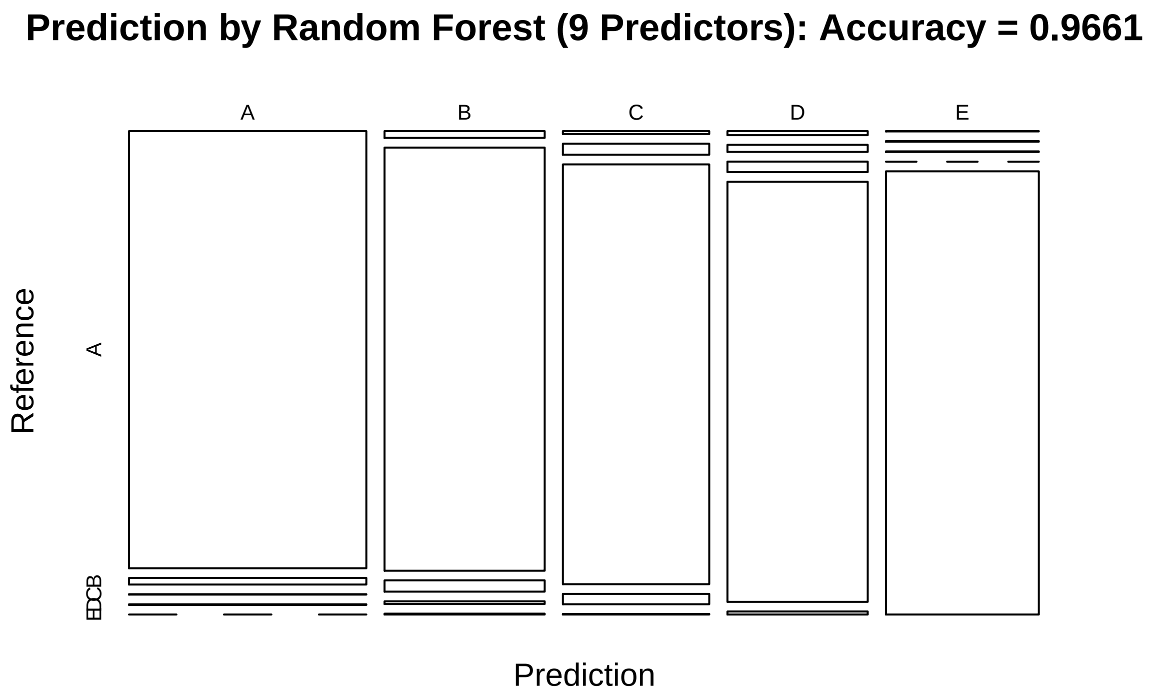
```r
# t7 <- Sys.time() # OPTIONAL SECTION
set.seed(3606)
library(doParallel)
cluster <- makeCluster(detectCores() - 1)
registerDoParallel(cluster)
# define control using random forest selection function
ctrl <- rfeControl(functions=rfFuncs, method="cv", number=10)
# run RFE algorithm
results <- rfe(training[, -32], training[, 32], sizes=c(1:15), rfeControl=ctrl)
stopCluster(cluster)
# print(results) # list chosen features # predictors(results)
# dim(training)
# t8 <- Sys.time(); t8 - t7 ## Time difference of 16.46114 mins
# t9 <- Sys.time() # OPTIONAL SECTION
set.seed(3606)
library(doParallel)
cluster <- makeCluster(detectCores() - 1)
registerDoParallel(cluster)
myTraining2 <- myTraining[, predictors(results)[1:15]]
myTraining2$classe <- myTraining$classe
myTesting2 <- myTesting[, predictors(results)[1:15]]
myTesting2$classe <- myTesting$classe
# dim(myTraining2); dim(myTesting2)
fitControl <- trainControl(method = "cv", number = 10, allowParallel = TRUE)
modFitRF2 <- train(classe ~ ., data=myTraining2, method = "rf",
                   trControl = fitControl)
predictionRF2 <- predict(modFitRF2, myTesting2, type = "raw")
ConfusionMatrixRF2 <- confusionMatrix(predictionRF2, myTesting2$classe)
# plot(modFitRF2, main="Final Model with Random Forests")
plot(ConfusionMatrixRF2$table, col = ConfusionMatrixRF2$byClass,
     main = paste("Prediction by Random Forest (15 Predictors): Accuracy =",
                  round(ConfusionMatrixRF2$overall['Accuracy'], 4)))
```



```r
stopCluster(cluster)
# t10 <- Sys.time(); t10 - t9 ## Time difference of 2.161987 mins
```

Predict Utilizing Random Forest plus Variable Selection by Importance

```r
# t11 <- Sys.time() #OPTIONAL SECTION
set.seed(3606)
library(doParallel)
cluster <- makeCluster(detectCores() - 1)
registerDoParallel(cluster)
A <- modFitRF$importance[, "MeanDecreaseAccuracy"]
B <- order(A, decreasing = TRUE); # A[B]
# "yaw_belt","roll_forearm, magnet_dumbbell_z", "roll_dumbbell", "pitch_forearm"
# "magnet_belt_y", "accel_forearm_x", "yaw_dumbbell", "total_accel_dumbbell"
myTraining3 <- myTraining[, B[1:9]]
myTraining3$classe <- myTraining$classe
myTesting3 <- myTesting[, B[1:9]]
myTesting3$classe <- myTesting$classe
fitControl <- trainControl(method = "cv", number = 10, allowParallel = TRUE)
modFitRF3 <- train(classe ~ ., data=myTraining3, method = "rf", trControl = fitControl)
predictionRF3 <- predict(modFitRF3, myTesting3, type = "raw")
ConfusionMatrixRF3 <- confusionMatrix(predictionRF3, myTesting3$classe)
# plot(modFitRF3, main="Final Model with Random Forests (9 Predictors)")
plot(ConfusionMatrixRF3$table, col = ConfusionMatrixRF3$byClass,
     main = paste("Prediction by Random Forest (9 Predictors): Accuracy =",
                  round(ConfusionMatrixRF3$overall['Accuracy'], 4)))
```



```
stopCluster(cluster)
# t12 <- Sys.time(); t12 - t11 ## Time difference of 1.490645 mins
```

"Repeated Cross Validation" is included in GBM modeling shown above (method='repeatedcv'). Also, in the optional sections, 10-fold Cross-Validation is used for Random Forest modeling.

Predict Test Data Set

When train data with myTraining, and predict outcome of myTesting, the accuracy of various methods are:

1. Decision Tree = 67.44%;
2. Random Forests = 99.17%;
3. Generalized Boosted Regression = 94.72%.

Therefore, **Random Forest** method will be used to make prediction here. Thus, the **expected out-of-sample error** is 100% - 99.17% = **0.83%**.

The Prediction Results are listed as below

```
set.seed(3606)
# predict with 31 variables
predictionTEST <- predict(modFitRF, testing, type = "class")
predictionTEST
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  B  A  A  E  D  B  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

```
# predict with 15 varibles selected by RFE
testing2 <- testing[, predictors(results)[1:15]]
predictionTEST2 <- predict(modFitRF2, testing2, type = "raw")
predictionTEST2
```

```
##  [1] B A B A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```

```
# predict with 9 variables slected by varImp
testing3 <- testing[, B[1:9]]
predictionTEST3 <- predict(modFitRF3, testing3, type = "raw")
predictionTEST3
```

```
##  [1] D A B C A E D B A A B C B A E E A B B B
## Levels: A B C D E
```

Summary

1. From original 159 variables, **31 variables** are selected to predict variable "classe".

2. Following techniques have been used to clean up the data set

- remove NearZeroVariance variables
- remove variables which contain more than 55% NA
- remove irrelavent variables
- remove highly correlated variables

3. Among three algorithm (Decision Tree, Random Forest and Garident Boosting Machine) investigated, **Random Forest** perform better, with which **99.17% accuracy** is obtained.
4. **Cross Validation** is employed inside the prediction method **through R package**.
5. A prediction for "testing" data set has been made, shown in section "Predict Test Data Set".
6. Optionally, **RFE** method used to select **15 predictors**, the Radon Forest **accuracy is 98.81%**.
7. optionally, **varImp** method used to select **9 predictors**, the Radon Forest **accuracy is 96.61%**.

Appendix

```r
cat(Sys.info()[1:2], "    ", R.version.string)
```

```
## Windows 10 x64     R version 3.2.3 (2015-12-10)
```