Prediction Assignment Writeup

Mathias Stein

June 21st 2018

Introduction

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Model built

The expacted outcome variable is classe, a 5 level of factor variable. In this dataset, participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different fashion: Class A - excatly according to the specification, Class B - throwing the elbows to the front, Class c- lifting the dumbbell only halfway, Class D - lowering the dumbbell only halfway, Class E - throwing the hips to the front.

Class A correspons to the specified execution of the exercise, while the other 4 classes correpond to common mistakes. Decision tree will be used to create the model. After the model have been developed. Cross-validation will be performed. Two set of data will be created, original training data set (75%) and subtesting data set (25%).

Load library

```
## Loading required package: lattice

## Loading required package: ggplot2

## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

Download and loading the Dataset

```
# Download the dataset
trainUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"


# Load the dataset into memory
trainingData <- read.csv(url(trainUrl), na.strings = c("NA", "#DIV/0!", ""))
testingData <- read.csv(url(testUrl), na.strings = c("NA", "#DIV/0!", ""))
```

```r
#
trainingData <- trainingData[, colSums(is.na(trainingData)) == 0]
testingData <- testingData[, colSums(is.na(testingData)) == 0]

# Delete variables that are not related
trainingData <- trainingData[, -c(1:7)]
testingData <- testingData[, -c(1:7)]

# partioning the training set into two different dataset

traningPartitionData <- createDataPartition(trainingData$classe,  p = 0.7, list = F)
trainingDataSet <- trainingData[traningPartitionData, ]
testingDataSet <- trainingData[-traningPartitionData, ]
dim(trainingData); dim(testingDataSet)
```

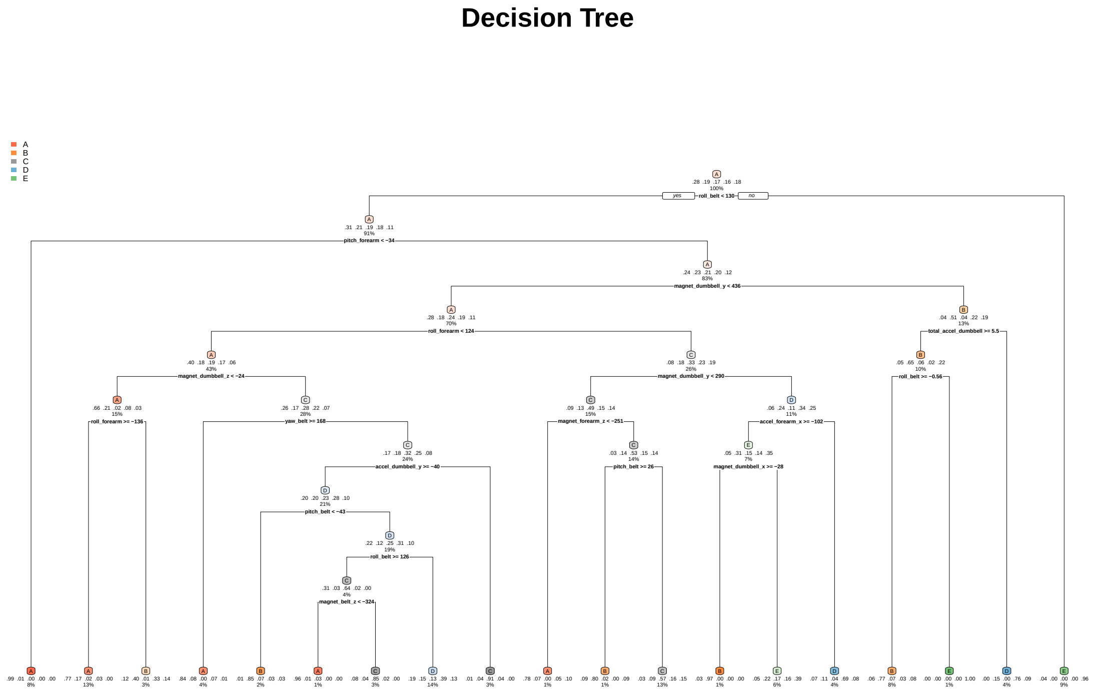
```
## [1] 19622    53
```

```
## [1] 5885   53
```

Prediction model 1 - decision tree

```r
decisionTreeModel <- rpart(classe ~ ., data = trainingDataSet, method = "class")
decisionTreePrediction <- predict(decisionTreeModel, testingDataSet, type = "class")

# Plot Decision Tree
rpart.plot(decisionTreeModel, main = "Decision Tree", under = T, faclen = 0)
```



```r
# Using confusion matrix to test results
confusionMatrix(decisionTreePrediction, testingDataSet$classe)
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1362  150   19   34    9
##          B   56  595   50   70   69
##          C   37   98  760  143  112
##          D  180  209  126  649  148
##          E   39   87   71   68  744
##
## Overall Statistics
##
##                Accuracy : 0.6984
##                  95% CI : (0.6865, 0.7101)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6202
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8136   0.5224   0.7407   0.6732   0.6876
## Specificity            0.9497   0.9484   0.9197   0.8653   0.9448
## Pos Pred Value         0.8653   0.7083   0.6609   0.4947   0.7374
## Neg Pred Value         0.9276   0.8922   0.9438   0.9311   0.9307
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2314   0.1011   0.1291   0.1103   0.1264
## Detection Prevalence   0.2675   0.1427   0.1954   0.2229   0.1715
## Balanced Accuracy      0.8816   0.7354   0.8302   0.7693   0.8162
```

Prediction model 2 - random forest

```
randomForestModel <- randomForest(classe ~. , data = trainingDataSet, method = "class")
randomForestPrediction <- predict(randomForestModel, testingDataSet, type = "class")

confusionMatrix(randomForestPrediction, testingDataSet$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    2    0    0    0
##          B    0 1134    5    0    0
##          C    0    3 1020    3    0
##          D    0    0    1  961    4
##          E    0    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.9969
##                  95% CI : (0.9952, 0.9982)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9961
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9956   0.9942   0.9969   0.9963
## Specificity            0.9995   0.9989   0.9988   0.9990   1.0000
## Pos Pred Value         0.9988   0.9956   0.9942   0.9948   1.0000
## Neg Pred Value         1.0000   0.9989   0.9988   0.9994   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1927   0.1733   0.1633   0.1832
## Detection Prevalence   0.2848   0.1935   0.1743   0.1641   0.1832
## Balanced Accuracy      0.9998   0.9973   0.9965   0.9979   0.9982
```

Prediction model 2 - random forest

From the result, it show Random Forest accuracy is higher than Decision tree which is 0.9915 > 0.6644. Therefore, we will use random forest to answer the assignment.

```
predictionFinal <- predict(randomForestModel, testingDataSet, type = "class")
#predictionFinal
```