Prediction Assignment

Thomas Schipritt

August 1, 2019

Project Goal

In this project, the goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to predict whether or not they're doing a dumbell curl correctly, using the classe variable in the data set.

Participants were asked to perform one set of 10 repetitionsof the Unilateral Dumbbell Biceps Curl in five different fash-ions: exactly according to the specification (Class A), throw-ing the elbows to the front (Class B), lifting the dumbbellonly halfway (Class C), lowering the dumbbell only halfway(Class D) and throwing the hips to the front (Class E).

Step 1 - Getting Data

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.4
```

```
## Loading required package: lattice
```

```
## Warning: package 'lattice' was built under R version 3.4.4
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```r
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.4.4
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```r
set.seed(4116)

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

Step 2 - Cleaning and Splitting Data for Cross Validation

Removing irrelevant variables, near zero variance variables, and any variables with a NA% of > 90% then splitting the training set into a train and test set

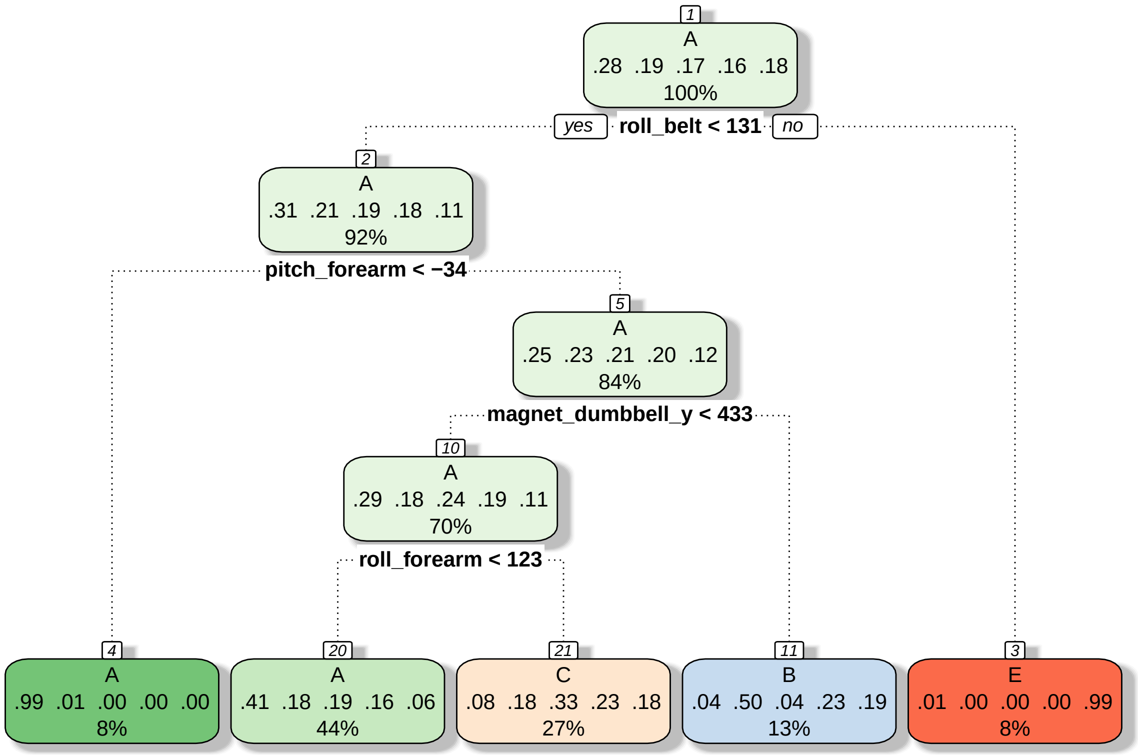
```r
training <- training[,-c(1:7)]
nzv <- nearZeroVar(training, saveMetrics = TRUE)
nzvvar <- rownames(nzv[nzv$nzv == TRUE,])
remove <- names(training) %in% nzvvar
training <- training[!remove]

narate <- data.frame(colSums(is.na(training))/nrow(training))
colnames(narate) <- "NArate"
narate$rnames <- rownames(narate)
hiNA <- narate$rnames[narate$NArate > .9]
removeNA <- names(training) %in% hiNA
training <- training[!removeNA]

intrain <- createDataPartition(y=training$classe, p =.75, list = FALSE)
modtrain <- training[intrain,]
modtest <- training[-intrain,]
```

Step 2 - Model Fit Tree and Accuracy

```r
rpartmodfit <- train(classe ~ ., data = modtrain, method = "rpart")
fancyRpartPlot(rpartmodfit$finalModel)
```



Rattle 2019−Aug−05 14:23:48 schiprt

```
rpartpredict <- predict(rpartmodfit, newdata=modtest)
confusionMatrix(rpartpredict, modtest$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1264  369  384  358  141
##          B   29  343   37  154  122
##          C   98  237  434  292  243
##          D    0    0    0    0    0
##          E    4    0    0    0  395
##
## Overall Statistics
##
##                Accuracy : 0.4967
##                  95% CI : (0.4826, 0.5108)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3428
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
```

```
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9061  0.36143   0.5076   0.0000  0.43840
## Specificity            0.6432  0.91353   0.7851   1.0000  0.99900
## Pos Pred Value         0.5024  0.50073   0.3328      NaN  0.98997
## Neg Pred Value         0.9451  0.85636   0.8831   0.8361  0.88768
## Prevalence             0.2845  0.19352   0.1743   0.1639  0.18373
## Detection Rate         0.2577  0.06994   0.0885   0.0000  0.08055
## Detection Prevalence   0.5131  0.13968   0.2659   0.0000  0.08136
## Balanced Accuracy      0.7746  0.63748   0.6464   0.5000  0.71870
```

The result is <50% accuracy, so let's try a different approach.

Step 3 - Model Fit Random Forest and Accuracy

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.4
```

```
## randomForest 4.6-14
```

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

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:rattle':
##
##     importance
```

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

```r
rfmodfit <- randomForest(classe ~., data = modtrain)
rfpredict <- predict(rfmodfit, newdata = modtest)
confusionMatrix(rfpredict, modtest$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    6    0    0    0
##          B    0  939    3    0    0
##          C    0    4  852    9    0
##          D    0    0    0  794    1
##          E    0    0    0    1  900
##
## Overall Statistics
```

```
##
##                Accuracy : 0.9951
##                  95% CI : (0.9927, 0.9969)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9938
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9895   0.9965   0.9876   0.9989
## Specificity            0.9983   0.9992   0.9968   0.9998   0.9998
## Pos Pred Value         0.9957   0.9968   0.9850   0.9987   0.9989
## Neg Pred Value         1.0000   0.9975   0.9993   0.9976   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1915   0.1737   0.1619   0.1835
## Detection Prevalence   0.2857   0.1921   0.1764   0.1621   0.1837
## Balanced Accuracy      0.9991   0.9944   0.9966   0.9937   0.9993
```

```r
print(rfmodfit)
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = modtrain)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.5%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4179    4    0    1    1 0.001433692
## B   14 2830    4    0    0 0.006320225
## C    0   13 2553    1    0 0.005453837
## D    0    0   26 2385    1 0.011194030
## E    0    0    0    8 2698 0.002956393
```

The random forest result is >99% accuracy. The out of estimate error rate is .51%. We would expect the test data to be slightly higher than that do to model over fitting.