Machine Learning Week 3 Assignment Exercise Prediction

Richard Ellison

October 22, 2015

Background and Summary

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 to predict the manner in which they did the exercise, using the classe variable in the training set. They you will use the prediction model built to predict 20 different test cases in different files for submission.

Data Processing

Read the Groupware@LES data, Human Activity Recognition

```
trainData <- read.csv("C:/Users/rellison/Documents/Coursera/MachineLearning/pml-training.csv")

testData <- read.csv("C:/Users/rellison/Documents/Coursera/MachineLearning/pml-testing.csv")
```

investigate the data

```
dim(trainData)
```

```
## [1] 19622   160
```

Classe variable is the outcome for prediction. Training data contains 19,622 observations and 160 variables.

```
dim(testData)
```

```
## [1]  20 160
```

Testing records contain 20 observations and 160 variables.

Subset the data to retain only the fields needed for the analysis # Since we only care about readings for belt, forearm, arm and dumbell, # remove missing values and fields with timestamp or window in the names

```
trainData <- trainData[, colSums(is.na(trainData))==0]
testData <- testData[, colSums(is.na(testData))==0]

classe<-trainData$classe
trainRemove <- grepl("^X|timestamp|window", names(trainData))
trainData <- trainData[, !trainRemove]
trainClean <- trainData[, sapply(trainData, is.numeric)]
trainClean$classe <- classe
head(trainClean)
```

```
##   roll_belt pitch_belt yaw_belt total_accel_belt gyros_belt_x gyros_belt_y
## 1      1.41       8.07    -94.4                3         0.00         0.00
## 2      1.41       8.07    -94.4                3         0.02         0.00
## 3      1.42       8.07    -94.4                3         0.00         0.00
## 4      1.48       8.05    -94.4                3         0.02         0.00
## 5      1.48       8.07    -94.4                3         0.02         0.02
## 6      1.45       8.06    -94.4                3         0.02         0.00
##   gyros_belt_z accel_belt_x accel_belt_y accel_belt_z magnet_belt_x
## 1        -0.02          -21            4           22            -3
## 2        -0.02          -22            4           22            -7
## 3        -0.02          -20            5           23            -2
## 4        -0.03          -22            3           21            -6
## 5        -0.02          -21            2           24            -6
## 6        -0.02          -21            4           21             0
##   magnet_belt_y magnet_belt_z roll_arm pitch_arm yaw_arm total_accel_arm
## 1           599          -313     -128      22.5    -161              34
## 2           608          -311     -128      22.5    -161              34
## 3           600          -305     -128      22.5    -161              34
## 4           604          -310     -128      22.1    -161              34
## 5           600          -302     -128      22.1    -161              34
## 6           603          -312     -128      22.0    -161              34
##   gyros_arm_x gyros_arm_y gyros_arm_z accel_arm_x accel_arm_y accel_arm_z
## 1        0.00        0.00       -0.02        -288         109        -123
## 2        0.02       -0.02       -0.02        -290         110        -125
## 3        0.02       -0.02       -0.02        -289         110        -126
## 4        0.02       -0.03        0.02        -289         111        -123
## 5        0.00       -0.03        0.00        -289         111        -123
## 6        0.02       -0.03        0.00        -289         111        -122
##   magnet_arm_x magnet_arm_y magnet_arm_z roll_dumbbell pitch_dumbbell
## 1         -368          337          516      13.05217      -70.49400
## 2         -369          337          513      13.13074      -70.63751
## 3         -368          344          513      12.85075      -70.27812
## 4         -372          344          512      13.43120      -70.39379
## 5         -374          337          506      13.37872      -70.42856
## 6         -369          342          513      13.38246      -70.81759
##   yaw_dumbbell total_accel_dumbbell gyros_dumbbell_x gyros_dumbbell_y
## 1    -84.87394                  37                0            -0.02
```

```
## 2    -84.71065                  37                0            -0.02
## 3    -85.14078                  37                0            -0.02
## 4    -84.87363                  37                0            -0.02
## 5    -84.85306                  37                0            -0.02
## 6    -84.46500                  37                0            -0.02
##   gyros_dumbbell_z accel_dumbbell_x accel_dumbbell_y accel_dumbbell_z
## 1             0.00             -234               47             -271
## 2             0.00             -233               47             -269
## 3             0.00             -232               46             -270
## 4            -0.02             -232               48             -269
## 5             0.00             -233               48             -270
## 6             0.00             -234               48             -269
##   magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z roll_forearm
## 1              -559               293               -65         28.4
## 2              -555               296               -64         28.3
## 3              -561               298               -63         28.3
## 4              -552               303               -60         28.1
## 5              -554               292               -68         28.0
## 6              -558               294               -66         27.9
##   pitch_forearm yaw_forearm total_accel_forearm gyros_forearm_x
## 1         -63.9        -153                 36            0.03
## 2         -63.9        -153                 36            0.02
## 3         -63.9        -152                 36            0.03
## 4         -63.9        -152                 36            0.02
## 5         -63.9        -152                 36            0.02
## 6         -63.9        -152                 36            0.02
##   gyros_forearm_y gyros_forearm_z accel_forearm_x accel_forearm_y
## 1            0.00           -0.02             192             203
## 2            0.00           -0.02             192             203
## 3           -0.02            0.00             196             204
## 4           -0.02            0.00             189             206
## 5            0.00           -0.02             189             206
## 6           -0.02           -0.03             193             203
##   accel_forearm_z magnet_forearm_x magnet_forearm_y magnet_forearm_z
## 1            -215              -17              654              476
## 2            -216              -18              661              473
## 3            -213              -18              658              469
## 4            -214              -16              658              469
## 5            -214              -17              655              473
## 6            -215               -9              660              478
##   classe
## 1      A
## 2      A
## 3      A
## 4      A
## 5      A
## 6      A
```

```
testRemove <- grepl("^X|timestamp|window", names(testData))
testData <- testData[, !testRemove]
testClean <- testData[, sapply(testData, is.numeric)]
dim(testClean)
```

```
## [1] 20 53
```

Build 70/30 training/validation data sets from the clean training dataset

```
set.seed(23456)
inTrain <- createDataPartition(trainClean$classe, p=0.70, list=F)
trainData2 <- trainClean[inTrain, ]
testData2 <- trainClean[-inTrain, ]
dim(trainData2)
```

```
## [1] 13737    53
```

```
dim(testData2)
```

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

Fit a Random Forest predictive model with 5-fold cross validation

```
controlRf <- trainControl(method="cv", 5)
modelRf <- train(classe ~ ., data=trainData2,method="rf",trControl=controlRf, ntree=250)
modelRf
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10990, 10988, 10991, 10989, 10990
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9906090  0.9881190  0.002176371  0.002754290
##   27    0.9908275  0.9883959  0.001444133  0.001828245
##   52    0.9851489  0.9812099  0.003393169  0.004296789
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

** Estimate the performance of the model on the validation data **

```
predictRf <- predict(modelRf, testData2)
confusionMatrix(testData2$classe, predictRf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1668    3    2    0    1
##          B   11 1128    0    0    0
##          C    0    3 1022    1    0
```

```
##          D    0    1    7  956    0
##          E    0    0    3    3 1076
##
## Overall Statistics
##
##                Accuracy : 0.9941
##                  95% CI : (0.9917, 0.9959)
##     No Information Rate : 0.2853
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9925
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9934   0.9938   0.9884   0.9958   0.9991
## Specificity            0.9986   0.9977   0.9992   0.9984   0.9988
## Pos Pred Value         0.9964   0.9903   0.9961   0.9917   0.9945
## Neg Pred Value         0.9974   0.9985   0.9975   0.9992   0.9998
## Prevalence             0.2853   0.1929   0.1757   0.1631   0.1830
## Detection Rate         0.2834   0.1917   0.1737   0.1624   0.1828
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9960   0.9958   0.9938   0.9971   0.9989
```

```
accuracy <- postResample(predictRf, testData2$classe)
accuracy
```

```
##  Accuracy     Kappa
## 0.9940527 0.9924767
```

```
ooserror <- 1 - as.numeric(confusionMatrix(testData2$classe,predictRf)$overall[1])
ooserror
```

```
## [1] 0.005947324
```

The estimated accuracy of the model is 99% and the estimated out-of-sample error is .6%

** Apply the model to the original test dataset **

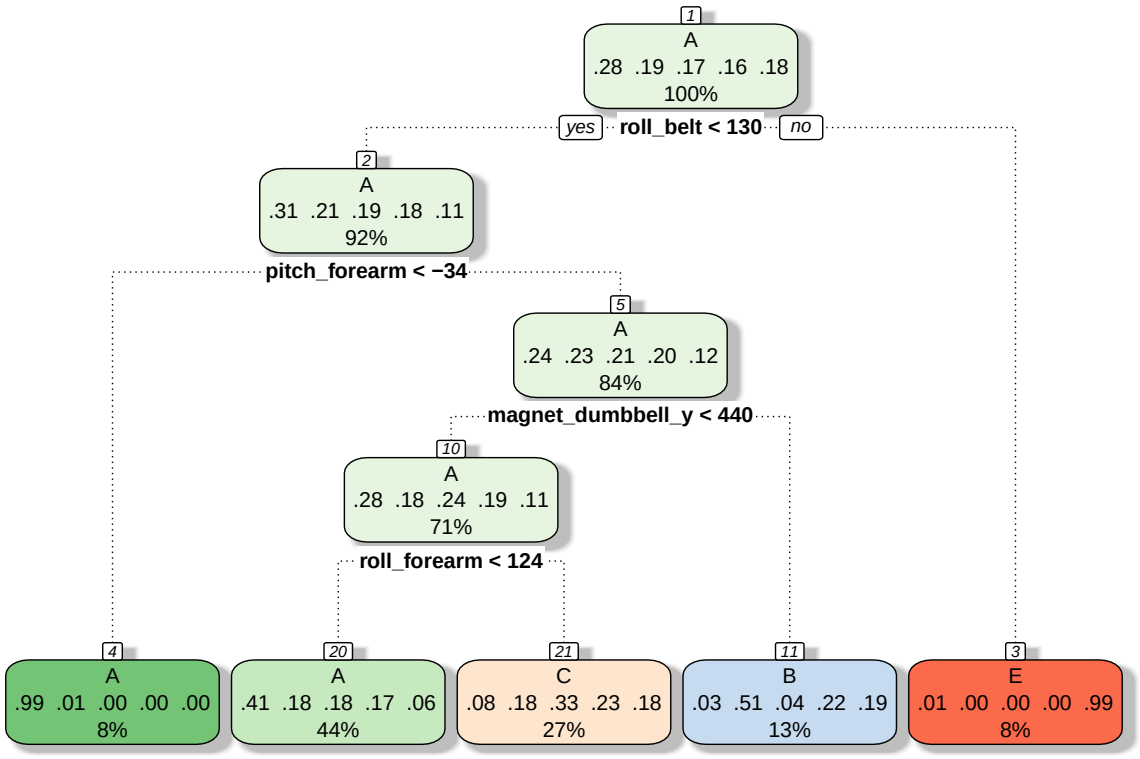
```
appliedModel <- predict(modelRf, testClean[, -length(names(testClean))])
appliedModel
```

```
##  [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
```

Appendix

** Show the fancy tree model for the original training classe variable**

```
FRPmodel <- train(classe ~ ., data=trainClean, method="rpart")
fancyRpartPlot(FRPmodel$finalModel)
```



Rattle 2015−Oct−23 17:33:53 rellison