Barbell Lift Analysis

Keelan Yang

April 10, 2017

Executive Summary

The purpose of this analysis is to predict how a person performed one of five barbell lifts based on the data from the Weight Lifting Exercise Dataset sourced from http://groupware.les.inf.puc-rio.br/har. The data are collected from wearable devices with accelerometers on the belt, forearm, arm, and dumbell from six participants. The resulting model will be used to predict 20 different test cases.

Data Preparation and Preprocessing

The initial training dataset contained 160 variables with 19622 observations, while the test dataset contained 160 variables with 20 observations.

```
##   dataset   obs variables
## 1   Train 19622       160
## 2    Test    20       160
```

After the data were loaded from the website, extraneous variables (e.g. user name, timestamps,etc.) were first removed. To reduce the number of potential predictors, analysis was performed on the test data set to remove variables that contained NAs. These variables were removed from both the training and test data sets.

```
## NACheck_Test
## FALSE  TRUE
##    53   100
```

Below is the list of 52 potential predictors that contained no NAs:

```
##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"
```

Next step is to create a validation data set from the training data set to allow for cross validation and to estimate the out-of-sample error rate.

```
# Create validation data set from training data set for
# cross-validation
set.seed(1100)
inTrain <- createDataPartition(y = traindata$classe, p = 0.7, list = FALSE)
validationdata <- traindata[-inTrain, ]
traindata <- traindata[inTrain, ]
```

After the validation data set was created, the training set was preprocessed using a center and scale methodology. This preprocessing was then applied to both the validation and test data sets.

```
set.seed(100)
prepro_train <- preProcess(traindata[, -which(names(traindata) %in%
    c("classe"))], method = c("center", "scale"))
traindata_PP <- predict(prepro_train, newdata = traindata[, -which(names(traindata) %in%
    c("classe"))])
traindata_PP <- data.frame(classe = traindata$classe, traindata_PP)

validationdata_PP <- predict(prepro_train, newdata = validationdata[,
    -which(names(validationdata) %in% c("classe"))])
validationdata_PP <- data.frame(classe = validationdata$classe, validationdata_PP)

testdata_PP <- predict(prepro_train, newdata = testdata)
```

Model Development

Given that the objective is to classify five potential exercise behaviors, both decision tree and random forest approaches were compared. First the models were developed on the training data set and then applied to the validation data set to quantify the potential out of sample error rates.

```
## Random Forest
set.seed(100)
fitRF <- randomForest(classe ~ ., data = traindata_PP)
predRF_train <- predict(fitRF, newdata = traindata_PP)

## Decision Tree
set.seed(100)
fitRpart <- train(classe ~ ., data = traindata_PP, method = "rpart")
predRpart_train <- predict(fitRpart, newdata = traindata_PP)
```

```
# Apply model to validation data sets
predRF_valid <- predict(fitRF, newdata = validationdata_PP)
predRpart_valid <- predict(fitRpart, newdata = validationdata_PP)
```

Model Selection

Review of the error rates, both in-sample and out-of-sample, showed that the random forest model performed significantly better than the decision tree model.

```
error_RF_train <- 1 - confusionMatrix(predRF_train, traindata_PP$classe)$overall[1]
error_Rpart_train <- 1 - confusionMatrix(predRpart_train, traindata_PP$classe)$overall[1]
error_RF_valid <- 1 - confusionMatrix(predRF_valid, validationdata_PP$classe)$overall[1]
error_Rpart_valid <- 1 - confusionMatrix(predRpart_valid, validationdata_PP$classe)$overall[1]
```

```
results_accuracy <- data.frame(method = c("Random Forest", "Decision Tree"),
    In_Sample = c(error_RF_train, error_Rpart_train), Out_Sample = c(error_RF_valid,
        error_Rpart_valid))
print(results_accuracy)
```

```
##          method In_Sample  Out_Sample
## 1 Random Forest 0.0000000 0.006627018
## 2 Decision Tree 0.5053505 0.505352591
```

The out-of-sample estimated error for the selected random forest model is **0.66%**. The decision tree model exhibited error rates ~50%.

20 Test Case Prediction

Below is the prediction for the 20 test cases provided using the selected random forest model.

```
pred_test <- predict(fitRF, newdata = testdata_PP)
print(pred_test)
```

```
##  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
```