Practical Machine Learning Project

Paul Hickman

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.

The goal of this project is to use data from accelerometers on the belt, forearm, arm, and dumbell to predict the activity 20 test subjects were performing.

Libraries

Load the required libraries.

```
library(caret)
library(rpart)
library(randomForest)
library(gbm)
library(plyr)
library(dplyr)
```

Data

Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E).

Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. 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).

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

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

Loading the Data

```
# get the training and test data
if (!file.exists("pml-training.csv"))
{
    download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",destfile="pml-
```

```
}
if (!file.exists("pml-testing.csv"))
{
    download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",destfile="pml-t
}

# load the training and test datasets (removing NA and error values)
training = read.csv("pml-training.csv", na.strings = c("", "NA", "#DIV/0!"))
testing = read.csv("pml-testing.csv", na.strings = c("", "NA", "#DIV/0!"))
```

Tidying the Data

Columns with little or no variability aren't good predictors and will be removed along with the first 7 columns that don't provide any useful information in our analysis.

```
# remove columns with no data
training = training[,colSums(is.na(training)) == 0]
testing = testing[,colSums(is.na(testing)) == 0]

# the first 7 columns aren't required for our analysis
training = training[, -c(1:7)]
testing = testing[, -c(1:7)]
```

Dividing the Data

Now the data has been loaded and in a useable state, the training dataset will be split into a training (60%) and validation (40%) dataset for the purpose of **cross-validation**.

```
set.seed(9243)
inTrain <- createDataPartition(training$classe, p = 0.6, list = FALSE)
validation <- training[-inTrain, ]
training <- training[inTrain, ]
```

Building the Models

The models will be built using all available variables in the dataset. The models to be created are classification tree, random forest and generalised boosted regression. These models have been selected based on their accuracy with this type of data. The models will be used against the validation dataset and the model with the highest accuracy will be used with the testing dataset.

Classification Tree Model

```
# build model
mRp = rpart(classe ~ ., method = "class", data = training)

# predict on the validation dataset
pRp = predict(mRp, validation, type = "class")
```

```r
# confusion matrix (for accuracy)
cmRp <- confusionMatrix(pRp, validation$classe)
cmRp
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1934  309   14  119   50
##          B   59  847  179   68  151
##          C   62  177 1034  201  156
##          D  117  128  104  834  148
##          E   60   57   37   64  937
##
## Overall Statistics
##
##                Accuracy : 0.712
##                  95% CI : (0.7018, 0.722)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6348
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8665   0.5580   0.7558   0.6485   0.6498
## Specificity            0.9124   0.9278   0.9080   0.9242   0.9660
## Pos Pred Value         0.7972   0.6495   0.6344   0.6266   0.8113
## Neg Pred Value         0.9450   0.8974   0.9463   0.9306   0.9245
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2465   0.1080   0.1318   0.1063   0.1194
## Detection Prevalence   0.3092   0.1662   0.2077   0.1696   0.1472
## Balanced Accuracy      0.8894   0.7429   0.8319   0.7864   0.8079
```

The accuracy of a classification tree model is **0.7119551**.

Random Forest Model

```r
# build model
mRf = randomForest(classe ~ ., data = training)

# predict on the validation dataset
pRf = predict(mRf, validation)

# confusion matrix (for accuracy)
cmRf <- confusionMatrix(pRf, validation$classe)
cmRf
```

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

```
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2229    9    0    0    0
##          B    2 1506   15    0    0
##          C    0    3 1350   22    3
##          D    0    0    3 1263    4
##          E    1    0    0    1 1435
##
## Overall Statistics
##
##                Accuracy : 0.992
##                  95% CI : (0.9897, 0.9938)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9898
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9987   0.9921   0.9868   0.9821   0.9951
## Specificity            0.9984   0.9973   0.9957   0.9989   0.9997
## Pos Pred Value         0.9960   0.9888   0.9797   0.9945   0.9986
## Neg Pred Value         0.9995   0.9981   0.9972   0.9965   0.9989
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2841   0.1919   0.1721   0.1610   0.1829
## Detection Prevalence   0.2852   0.1941   0.1756   0.1619   0.1832
## Balanced Accuracy      0.9985   0.9947   0.9913   0.9905   0.9974
```

The accuracy of a random forest model is **0.9919704**.

Generalised Boosted Regression Model

```
# build model
# v. slow and similar accuracy
mGbm = train(classe~., data=training, method="gbm", verbose=FALSE,
             trControl=trainControl(method="repeatedcv",number=5,repeats=1))

# predict on the validation dataset
pGbm = predict(mGbm, validation)

# confusion matrix (for accuracy)
cmGbm <- confusionMatrix(pGbm, validation$classe)
cmGbm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2201   60    0    2    1
```

```
##          B   20 1407   47    9   21
##          C    6   47 1304   36   16
##          D    4    0   15 1229   20
##          E    1    4    2   10 1384
##
## Overall Statistics
##
##                Accuracy : 0.9591
##                  95% CI : (0.9545, 0.9634)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9482
##  Mcnemar's Test P-Value : 4.249e-11
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9861   0.9269   0.9532   0.9557   0.9598
## Specificity            0.9888   0.9847   0.9838   0.9941   0.9973
## Pos Pred Value         0.9722   0.9355   0.9255   0.9692   0.9879
## Neg Pred Value         0.9944   0.9825   0.9901   0.9913   0.9910
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2805   0.1793   0.1662   0.1566   0.1764
## Detection Prevalence   0.2886   0.1917   0.1796   0.1616   0.1786
## Balanced Accuracy      0.9874   0.9558   0.9685   0.9749   0.9786
```

The accuracy of a boosted model is **0.9590874**.

Model Selection

The **random forest model** has the highest accuracy and will be selected. The out of sample error is calculated by 1 - accuracy, which in this case is **0.0080296**.

Testing

The random forest model will be run on the test dataset to predict the category of exercise being performed.

```
# the predict function is not happy with the data types as they are slightly different
# copying in a row of data from the training data and then removing it was the best
# way to have the testing match the training data types
testing_new = rbind.fill(training[2, -ncol(training)], testing)
testing_new <- testing_new[-1, ]
```

The answer to the Coursera project is:

```
# predict on the testing dataset
pTest = predict(mRf, testing_new)
pTest
```

```
##  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21
##  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
```