ClassProject

GG

8/5/2020

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. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

Executive Summary

The goal of this project is to predict the manner in which six participants did the exercise. This is the "classe" variable in the training set.

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). These are the categories for the "classe" variable.

Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. Participants were supervised by an experienced weight lifter to make sure the execution complied to the manner they were supposed to simulate. The exercises were performed by six male participants aged between 20-28 years, with little weight lifting experience. We made sure that all participants could easily simulate the mistakes in a safe and controlled manner by using a relatively light dumbbell (1.25kg).

Main Analysis

Importing and Cleaning the Data

```
##Importing the libraries for the analysis
library(utils)
library(caret)
library(randomForest)
library(ggplot2)
library(rpart)
```

```
fileURLtrain <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
fileURLtest <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

## Marking all missing data as NA in the data
training <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!",""))
testing <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!",""))

## Checking the dimentions of both datasets
dim(training)
```

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

```
dim(testing)
```

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

```
## Next we remove the variables that have no recorded value (all missing)
training <- training[,colSums(is.na(training)) == 0]
testing <- testing[,colSums(is.na(testing)) == 0]

## We can also remove the variabes that are not necessary of our analysis
training <- training[,-c(1:7)]
testing <- testing[,-c(1:7)]
```

Before cleaning
Training dataset has 19622 observations with 160 variables
Testing dataset has 20 observations with 160 variables

```
## Checking the dimentions of both datasets after cleaning them
dim(training)
```

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

```
dim(testing)
```

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

After cleaning
Training dataset has 19622 observations with 53 variables
Testing dataset has 20 observations with 53 variables

```
## Partitioning the training dataset with 70% for training and 30% testing
inTrain <- createDataPartition(y=training$classe, p=0.70, list=FALSE)
training_train <- training[inTrain, ]
training_test <- training[-inTrain, ]
dim(training_train)
```

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

```
dim(training_test)
```

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

I partition the training dataset with 70% for training and 30% for testing.

Thus, we get 13737 observations for training, 5885 observations for testing subsets.

Checking the frequency of classe variable's categories

As we mentioned earlier the variable consists of 5 categories and while category A has the highest frequency the other categories appear at a similar rate.

```
##
##    A    B    C    D    E
## 3906 2658 2396 2252 2525
```

Predictions and Cross Validation

Random Forest and Decision Tree models are used for training purposes.

```
mod1 <- train(classe ~., method ="rpart", data = training_train)
        ## Decision Tree model
pred1 <- predict(mod1, training_test)
mod2 <- randomForest(classe ~., type = "class", data=training_train)
        ## Random forest model
pred2 <- predict(mod2, training_test)
```

Now we can check the accuracies of all the models.

```
## Model 1 - Decision tree
confusionMatrix(pred1, training_test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1539  467  486  424  179
```

```
##          B   36  399   39  167  157
##          C   90  273  501  373  288
##          D    0    0    0    0    0
##          E    9    0    0    0  458
##
## Overall Statistics
##
##                Accuracy : 0.4923
##                  95% CI : (0.4794, 0.5051)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3358
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9194   0.3503  0.48830   0.0000  0.42329
## Specificity            0.6305   0.9159  0.78926   1.0000  0.99813
## Pos Pred Value         0.4973   0.5000  0.32852      NaN  0.98073
## Neg Pred Value         0.9516   0.8545  0.87959   0.8362  0.88483
## Prevalence             0.2845   0.1935  0.17434   0.1638  0.18386
## Detection Rate         0.2615   0.0678  0.08513   0.0000  0.07782
## Detection Prevalence   0.5259   0.1356  0.25913   0.0000  0.07935
## Balanced Accuracy      0.7749   0.6331  0.63878   0.5000  0.71071
```

```
## Model 2 - Random Forest
confusionMatrix(pred2, training_test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    2    0    0    0
##          B    1 1137    3    0    0
##          C    0    0 1021    6    0
##          D    0    0    2  956    3
##          E    1    0    0    2 1079
##
## Overall Statistics
##
##                Accuracy : 0.9966
##                  95% CI : (0.9948, 0.9979)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9957
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
```

```
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9988   0.9982   0.9951   0.9917   0.9972
## Specificity            0.9995   0.9992   0.9988   0.9990   0.9994
## Pos Pred Value         0.9988   0.9965   0.9942   0.9948   0.9972
## Neg Pred Value         0.9995   0.9996   0.9990   0.9984   0.9994
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2841   0.1932   0.1735   0.1624   0.1833
## Detection Prevalence   0.2845   0.1939   0.1745   0.1633   0.1839
## Balanced Accuracy      0.9992   0.9987   0.9969   0.9953   0.9983
```

As we can see the model using Random Forest method gives us the best predictor with model accuracy of 0.9913. The Decision tree model is in distant second place with the accuracy of 0.5593. Combining two predictors in this case did not give a better predictor. It's accuracy is only 0.4757. The expected out of sample error is 0.6% (1-0.9942). As a result, we will use the random forest model for our final prediction on validation data set.

Predicting the outcomes in the testing dataset

Now we can use your prediction model to predict 20 different test cases.

```
##  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
```