

Assignment Writeup - Prediction of how was Exercise Performed

Human activity recognition research has traditionally focused on discriminating between different activities. However, the “how (well)” investigation has only received little attention so far, even though it potentially provides useful information for a large variety of applications, such as sports training (<http://groupware.les.inf.puc-rio.br/har>).

For the prediction of how well individuals performed the assigned exercise 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).

This report aims to use machine learning algorithms to predict the class of exercise the individuals was performing by using measurements available from devices such as Jawbone Up, Nike FuelBand, and Fitbit.

Data Cleaning

The data for this project came was obtained from <http://groupware.les.inf.puc-rio.br/har>. Two data sets were available a training set and a test set for which 20 individuals without any classification for the class of exercise was available.

```
#Data loading

setwd("~/Desktop/coursera/R_courses/machine_learning")

pmlTrain<-read.csv("pml-training.csv", header=T, na.strings=c("NA", "#DIV/0!"))

pmlTest<-read.csv("pml-testing.csv", header=T, na.string=c("NA", "#DIV/0!"))
```

Training data was partitioned and preprocessed using the code described below. In brief, all variables with at least one “NA” were excluded from the analysis. Variables related to time and user information were excluded for a total of 51 variables and 19622 class measurements. Same variables were maintained in the test data set (Validation dataset) to be used for predicting the 20 test cases provided.

```
## NA exclusion for all available variables

noNAPmlTrain<-pmlTrain[, apply(pmlTrain, 2, function(x) !any(is.na(x)))]

dim(noNAPmlTrain)

## [1] 19622    60
```

```
## variables with user information, time and undefined
cleanpmlTrain<-noNApmlTrain[,-c(1:8)]

dim(cleanpmlTrain)

## [1] 19622    52

## 20 test cases provided clean info - Validation data set
cleanpmltest<-pmlTest[,names(cleanpmlTrain[, -52])]

dim(cleanpmltest)

## [1] 20 51
```

Data Partitioning and Prediction Process

The cleaned downloaded data set was subset in order to generate a test set independent from the 20 cases provided set. Partitioning was performed to obtain a 75% training set and a 25% test set.

```
#data cleaning
library(caret)

inTrain<-createDataPartition(y=cleanpmlTrain$classe, p=0.75,list=F)

training<-cleanpmlTrain[inTrain,]

test<-cleanpmlTrain[-inTrain,]

#Training and test set dimensions

dim(training)

## [1] 14718    52

dim(test)

## [1] 4904    52
```

Results and Conclusions

Random forest trees were generated for the training dataset using cross-validation. Then the generated algorithm was examined under the partitioned training set to examine the accuracy and estimated error of prediction. By using 51 predictors for five classes using cross-validation at a 5-fold an accuracy of 99.2% with a 95% CI [0.989-0.994] was achieved accompanied by a Kappa value of 0.99.

```
library(caret)

set.seed(13333)
```

```
fitControl2<-trainControl(method="cv", number=5, allowParallel=T, verbose=T)
rffit<-train(classe~.,data=training, method="rf", trControl=fitControl2, verbose=F)
```

```
## + Fold1: mtry= 2
```

```
## - Fold1: mtry= 2
```

```
## + Fold1: mtry=26
```

```
## - Fold1: mtry=26
```

```
## + Fold1: mtry=51
```

```
## - Fold1: mtry=51
```

```
## + Fold2: mtry= 2
```

```
## - Fold2: mtry= 2
```

```
## + Fold2: mtry=26
```

```
## - Fold2: mtry=26
```

```
## + Fold2: mtry=51
```

```
## - Fold2: mtry=51
```

```
## + Fold3: mtry= 2
```

```
## - Fold3: mtry= 2
```

```
## + Fold3: mtry=26
```

```
## - Fold3: mtry=26
```

```
## + Fold3: mtry=51
```

```
## - Fold3: mtry=51
```

```
## + Fold4: mtry= 2
```

```
## - Fold4: mtry= 2
```

```
## + Fold4: mtry=26
```

```
## - Fold4: mtry=26
```

```
## + Fold4: mtry=51
```

```
## - Fold4: mtry=51
```

```
## + Fold5: mtry= 2
```

```
## - Fold5: mtry= 2
```

```
## + Fold5: mtry=26
```

```
## - Fold5: mtry=26
## + Fold5: mtry=51
## - Fold5: mtry=51
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 26 on full training set
```

```
predrf<-predict(rffit, newdata=test)
confusionMatrix(predrf, test$classe)
```

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

```
##
```

```
##           Reference
## Prediction  A    B    C    D    E
##           A 1394    6    0    0    0
##           B    0  938    4    0    0
##           C    0    5  848    7    0
##           D    0    0    3  796    3
##           E    1    0    0    1  898
```

```
##
```

```
## Overall Statistics
```

```
##
```

```
##           Accuracy : 0.994
```

```
##           95% CI : (0.991, 0.996)
```

```
##           No Information Rate : 0.284
```

```
##           P-Value [Acc > NIR] : <2e-16
```

```
##
```

```
##           Kappa : 0.992
```

```
##           McNemar's Test P-Value : NA
```

```
##
```

```
## Statistics by Class:
```

```
##
##           Class: A Class: B Class: C Class: D Class: E
## Sensitivity           0.999   0.988   0.992   0.990   0.997
## Specificity           0.998   0.999   0.997   0.999   1.000
## Pos Pred Value        0.996   0.996   0.986   0.993   0.998
## Neg Pred Value        1.000   0.997   0.998   0.998   0.999
## Prevalence            0.284   0.194   0.174   0.164   0.184
## Detection Rate        0.284   0.191   0.173   0.162   0.183
## Detection Prevalence  0.285   0.192   0.175   0.164   0.184
## Balanced Accuracy     0.999   0.994   0.994   0.994   0.998
```

```
pred20<-predict(rffit, newdata=cleanpmltest)
# Output for the prediction of the 20 cases provided
pred20
```

```
## [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
```

A boosting algorithm was also run to confirm and be able to compare predictions. Data is not shown but the boosting approach presented less accuracy (96%) (Data not shown). However, when the predictions for the 20 test cases were compared match was same for both ran algorithms.

```
fitControl2<-trainControl(method="cv", number=5, allowParallel=T, verbose=T)
gmbfit<-train(classe~.,data=training, method="gbm", trControl=fitControl2, verbose=F)
gmbfit$finalModel
class(gmbfit)
predgmb<-predict(gmbfit, newdata=test)
confusionMatrix(predgmb, test$classe)
predtrain<-predict(gmbfit, newdata=training)
confusionMatrix(predtrain, training$classe)
predtrain<-predict(gmbfit, newdata=training)
confusionMatrix(predtrain, training$classe)
```

Once, the predictions were obtained for the 20 test cases provided, the below shown script was used to obtain single text files to be uploaded to the courses web site to comply with the submission assignment. 20 out of 20 hits also confirmed the accuracy of the obtained models.

```
getwd()

pml_write_files = function(x){

  n = length(x)

  for(i in 1:n){

    filename = paste0("problem_id_",i,".txt")

    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)

  }

}

pml_write_files(pred20)
```