Practical Machine Learning : Qualitative Prediction

H.Harvey

10 February 2016

Executive 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.

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, the goal is 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. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways:

- Class A: exactly according to the specification
- Class B: throwing the elbows to the front
- Class C: lifting the dumbbell only halfway
- Class D: lowering the dumbbell only halfway
- Class E: throwing the hips to the front

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).

Predicted model algorithm

To achieve the objective of this project, models are trained using the mentioned algorithms below since the objective is to predict a qualitative "Class" or category response.

- Recursive partitioning for classification & regression (Rpart)
- RandomForest (randomForest)

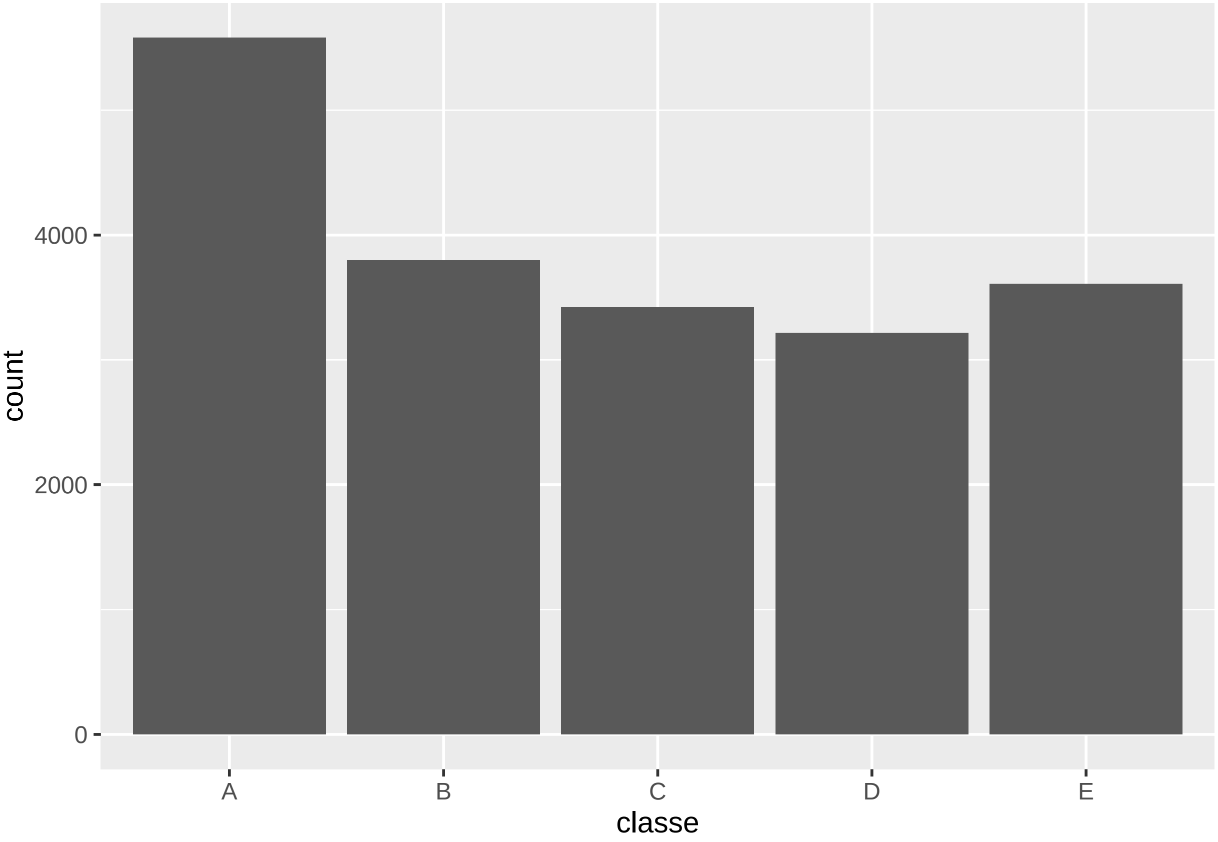
After repeated trials from our training dataset, we can conclude that the RandomForest algorithm offer a better strategy to predict the manner the "Class" are performing their exercices.

Data prepartion / procession

```r
setwd("/Users/hansharvey/Documents/Personnal Folder/Coursera/Data Science/Data Science Toolbox/Machine-
# After an inital file load, it has been observed that a lot of variable contained
# NA, blank space and #DIV/0! value.
# Easier and shortest solution to consider blank space, #DIV/0! and NA as NA
trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
validUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
train_file <- read.csv(url(trainUrl), na.strings = c('NA','#DIV/0!',''))
valid_file <- read.csv(url(validUrl), na.strings = c('NA','#DIV/0!',''))
```

An histogram of the frequency of shows the classe variable values across the dataset.

```r
ggplot(train_file, aes(classe, ..count..)) + geom_bar()
```



We can observe that the classe variable has five possible values; A, B, C, D, and E as defined in the introduction. the classe A represents the ideal weight-lifting method where the highest number of observations registered.

```r
# Remove column containing NA value (based on the parameter from the previous read.csv)
train_data <- train_file[,colSums(is.na(train_file)) == 0]
valid_data <- valid_file[,colSums(is.na(train_file)) == 0]
```

Based on the research paper by Velloso et al. (2013) p3. the first variables contained at the beginning of the dataset would not be required in the model prediction. Hence they are removed.

```r
train_data<-train_data[,-(1:7)]
valid_data<-valid_data[,-(1:7)]
```

Partitionning of the dataset in training and testing set.

To train our prediction model, the dataset was split into training and testing data using a 70/30 ratio.

```r
# Partition data set between training and validation
set.seed(101)
intrain <- createDataPartition(train_data$classe, p = 0.7, list = FALSE)
training = train_data[intrain, ] # 70% split
testing <- train_data[-intrain, ] # 30% split
dim(training);dim(testing)
```

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

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

We can observe that the training set hase 53 variables with 13737 observations and the testing set has the same number of variable but fewer obsservsation (5885).

Prediction model

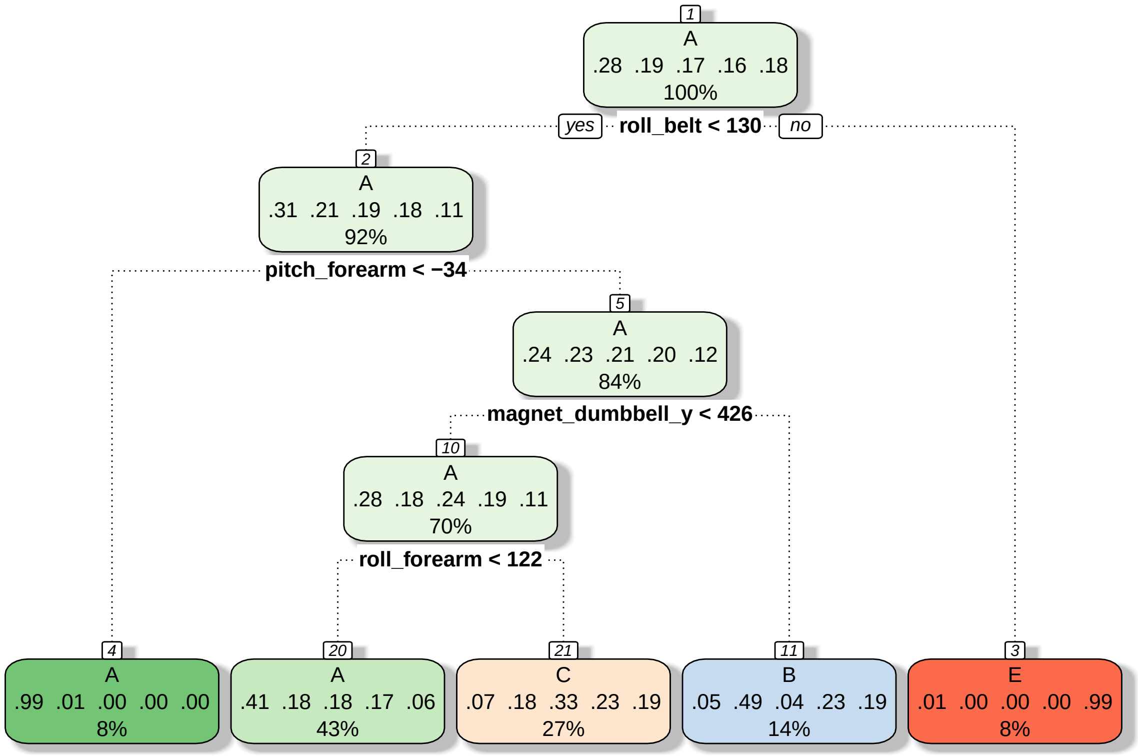
As mentioned earlier, the outcome (classe) is a categorical variable. In this direction, our algorithm should be based on one of those which are able to model categorical (Tree, boosting, bagging, randomForest).

Hence, a classification Tree algorithm was first used to test and predict qualitative outcome followed by a randomForest algorithm.

```r
# Recursive partitioning for classification & regression
set.seed(202)
# Limiting the number of bootstrap resample to 5 (instead of 25 as default)
# Using a Cross-valildation method
fitControl = trainControl(method = "cv", number = 5)
# The application of the rpart() generated more accuracy than the train() function
# but created more variances. Hence the selection of the train() function
# model_fit_Rpart <- rpart(classe ~ ., data=training, method="class")
model_fit_Rpart <- train(classe ~.,data = training,method="rpart",trControl=fitControl)
model_fit_Rpart$finalModel
```

```
## n= 13737
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 13737 9831 A (0.28 0.19 0.17 0.16 0.18)
##    2) roll_belt< 130.5 12614 8718 A (0.31 0.21 0.19 0.18 0.11)
##      4) pitch_forearm< -33.65 1111   10 A (0.99 0.009 0 0 0) *
##      5) pitch_forearm>=-33.65 11503 8708 A (0.24 0.23 0.21 0.2 0.12)
##       10) magnet_dumbbell_y< 426.5 9557 6852 A (0.28 0.18 0.24 0.19 0.11)
##         20) roll_forearm< 122.5 5912 3471 A (0.41 0.18 0.18 0.17 0.059) *
##         21) roll_forearm>=122.5 3645 2425 C (0.072 0.18 0.33 0.23 0.19) *
##       11) magnet_dumbbell_y>=426.5 1946  992 B (0.046 0.49 0.044 0.23 0.19) *
##    3) roll_belt>=130.5 1123   10 E (0.0089 0 0 0 0.99) *
```

```r
fancyRpartPlot(model_fit_Rpart$finalModel)
```



Rattle 2016−Feb−10 21:14:34 hansharvey

Evaluating the classification tree model

After training the model we compare its outcome to the testing data actual outcome using the confusion matrix function.

```
# Testing the model and predicting value from the testing set.
predict_Rpart<-predict(model_fit_Rpart,testing)
# predict_Rpart<-predict(model_fit_Rpart,testing, type='class')
conf_matrix_Rpart<-confusionMatrix(predict_Rpart,testing$classe)
conf_matrix_Rpart
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1503  464  477  433  147
##          B   33  394   44  178  151
##          C  134  281  505  353  266
##          D    0    0    0    0    0
##          E    4    0    0    0  518
##
## Overall Statistics
##
##                Accuracy : 0.4962
##                  95% CI : (0.4833, 0.509)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3419
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8978  0.34592  0.49220   0.0000  0.47874
## Specificity            0.6388  0.91445  0.78720   1.0000  0.99917
## Pos Pred Value         0.4970  0.49250  0.32814      NaN  0.99234
## Neg Pred Value         0.9402  0.85349  0.88012   0.8362  0.89483
## Prevalence             0.2845  0.19354  0.17434   0.1638  0.18386
## Detection Rate         0.2554  0.06695  0.08581   0.0000  0.08802
## Detection Prevalence   0.5138  0.13594  0.26151   0.0000  0.08870
## Balanced Accuracy      0.7683  0.63019  0.63970   0.5000  0.73896
```

We observe with the above model, an accuracy of 0.4962, which is not offering an acceptable level of performance to conclude that this model would generate accurate prediction.

The Random Forest algorithm was used to get a higher level of accuracy with the same numbers of predictor variables (52).

```
# Model Random Forest
set.seed(303)
# Setting the parameter for the resampling method to "out of Box" sampling and to limit the number of 5
trControl = trainControl(method = "oob", number=5)
# May take more than 10min
model_fit_RF <- train(classe ~.,data = training, method="rf", trControl=trControl)
model_fit_RF
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9923564  0.9903305
##   27    0.9924292  0.9904230
##   52    0.9836937  0.9793713
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

We observe that the Random Forest model resulted in 99.2% accurate (with mtry=27). Thus this would be a more efficient model to use in predicitng our data.

After training the model we compare its outcome to the testing data actual outcome using the onfusion matrix function.

```
# Testing the model and predicting value from the testing set.
predict_RF<-predict(model_fit_RF,testing)
conf_matrix_RF<-confusionMatrix(predict_RF,testing$classe)
conf_matrix_RF
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1670    9    0    0    0
##          B    2 1125    2    0    0
##          C    1    5 1022    7    3
##          D    0    0    2  954    6
##          E    1    0    0    3 1073
## 
## Overall Statistics
##                                          
##                Accuracy : 0.993          
##                  95% CI : (0.9906, 0.995)
##     No Information Rate : 0.2845         
##     P-Value [Acc > NIR] : < 2.2e-16      
##                                          
##                   Kappa : 0.9912         
##  Mcnemar's Test P-Value : NA             
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9976   0.9877   0.9961   0.9896   0.9917
## Specificity            0.9979   0.9992   0.9967   0.9984   0.9992
## Pos Pred Value         0.9946   0.9965   0.9846   0.9917   0.9963
## Neg Pred Value         0.9990   0.9971   0.9992   0.9980   0.9981
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2838   0.1912   0.1737   0.1621   0.1823
## Detection Prevalence   0.2853   0.1918   0.1764   0.1635   0.1830
## Balanced Accuracy      0.9977   0.9934   0.9964   0.9940   0.9954
```

Cross-Validation

```
#Test file validation
predict_valid <- predict(model_fit_RF, valid_data)
predict_valid
```

```
##  [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
```

Reference

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://groupware.les.inf.puc-rio.br/har

Velloso, E. and al. (2013) Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI, Germany