Practical Machine Learning - Assignment 1

Arthi Murugesan

March 27, 2016

Introduction

In this report, we will be analysing the personal activity of 6 participants, doing barbell lifts in 5 different ways. We will try to see given an accelerometer reading on the belt, forearm,arm and dumbell, which of the 5 lifts are being perfomed.

The dataset used for the analysis is available at http://groupware.les.inf.puc-rio.br/har and more information regarding the dataset is available at [1]

Exploratory Data Analysis

The data set consist of totally 160 values including the user name and classe. Let's take a deeper look at the 160 parameters provided to check if they all consist of non NA values. If there are any columns with no values provided anywhere, it's clear they add no value to be use in the model training process, so we can remove them from the training set. Similarly the columns which were not used in the training are not going to be useful in predictions, Hence can also be removed from the test set. Also, personal identifiers such as user name, or the timestamp does not add any value related to the activity, Hence they can also be removed from the training and test set.

There are totally 5 different types of practical activity that are captured and the classe encodes these differences. Our Models will be predicting these 5 different barbell lifts (classe), given the other parameters.

```
rm(list=ls())
training_url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testing_url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

download.file(training_url, destfile = "pml-training.csv", method="curl", mode="wb")
download.file(testing_url, destfile = "pml-testing.csv", method="curl", mode="wb")

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

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

dim(training_data)
```

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

```
dim(test_data)
```

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

```
summary(training_data$classe)
```

```
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

```r
summary(training_data$user_name)
```

```
##   adelmo carlitos  charles   eurico   jeremy    pedro
##     3892     3112     3536     3070     3402     2610
```

```r
# Removing Parameters which hold no value
col_sum <- colSums(is.na(training_data))
reduced_training_data <- training_data[,colSums(is.na(training_data))==0]
reduced_test_data <- test_data[,colSums(is.na(training_data))==0]

#Removing Parameters which are not related to classe
clean_training_data <- reduced_training_data[,-c(1:7)]
test <- reduced_test_data[,-c(1:7)]
```

Training

The training set is split into 75-25 for training and dev set. We will use cross validation in training and evaluate the model perfomance on the dev set. Finally once we have picked the model parameters using dev set, we will evaluate our model over the blind set, namely the test set (20 testcases) to see how the model performs. This follows the usual practice of model training, evaluation and testing.

```r
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```r
set.seed(123)
inTrain <- createDataPartition(clean_training_data$classe, p = .75, list = FALSE)
training <- clean_training_data[inTrain,]
dev_test <- clean_training_data[-inTrain,]
```

Decision Trees

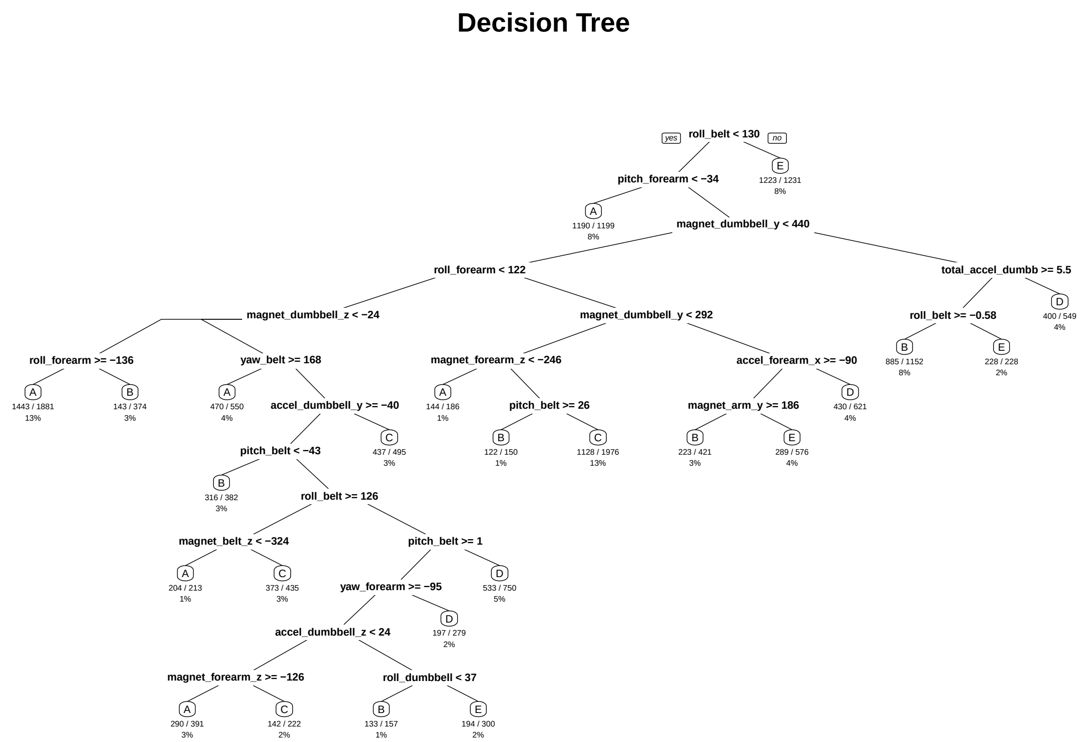
We will start with Decision Trees, as they are simple and parsimonious models.

```r
library(rpart)
library(rpart.plot)
#Cross Validation & Model Training

model_decisiontree <- rpart(classe ~ ., data=training, method="class")

# Predicting:
predict_dt <- predict(model_decisiontree, dev_test, type = "class")

# Plot of the Decision Tree
rpart.plot(model_decisiontree, main="Decision Tree", extra=102, under=TRUE, faclen=0)
```




```r
#confusion matrix
confusionMatrix(predict_dt, dev_test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1237  131   16   44   15
##          B   45  598   72   67   71
##          C   39  102  683  134  115
##          D   51   64   65  499   46
##          E   23   54   19   60  654
##
## Overall Statistics
##
##                Accuracy : 0.7486
##                  95% CI : (0.7362, 0.7607)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6817
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8867   0.6301   0.7988   0.6206   0.7259
## Specificity            0.9413   0.9355   0.9037   0.9449   0.9610
## Pos Pred Value         0.8572   0.7011   0.6365   0.6883   0.8074
## Neg Pred Value         0.9543   0.9134   0.9551   0.9270   0.9397
```

```
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2522   0.1219   0.1393   0.1018   0.1334
## Detection Prevalence   0.2942   0.1739   0.2188   0.1478   0.1652
## Balanced Accuracy      0.9140   0.7828   0.8513   0.7828   0.8434
```

The decision trees have a prediction accuracy of 74.86% of the dev set.According to the model,roll_belt seems to the main predictive parameter.

Random Forest

Following up on the simple decision tree model perfomance, random forest will be used to gain some ground. As for the cross validation, we will use Out-of-Bag cross validation method as it is specifically good for random forests (also useful with bagged trees, condition tree forest models etc).

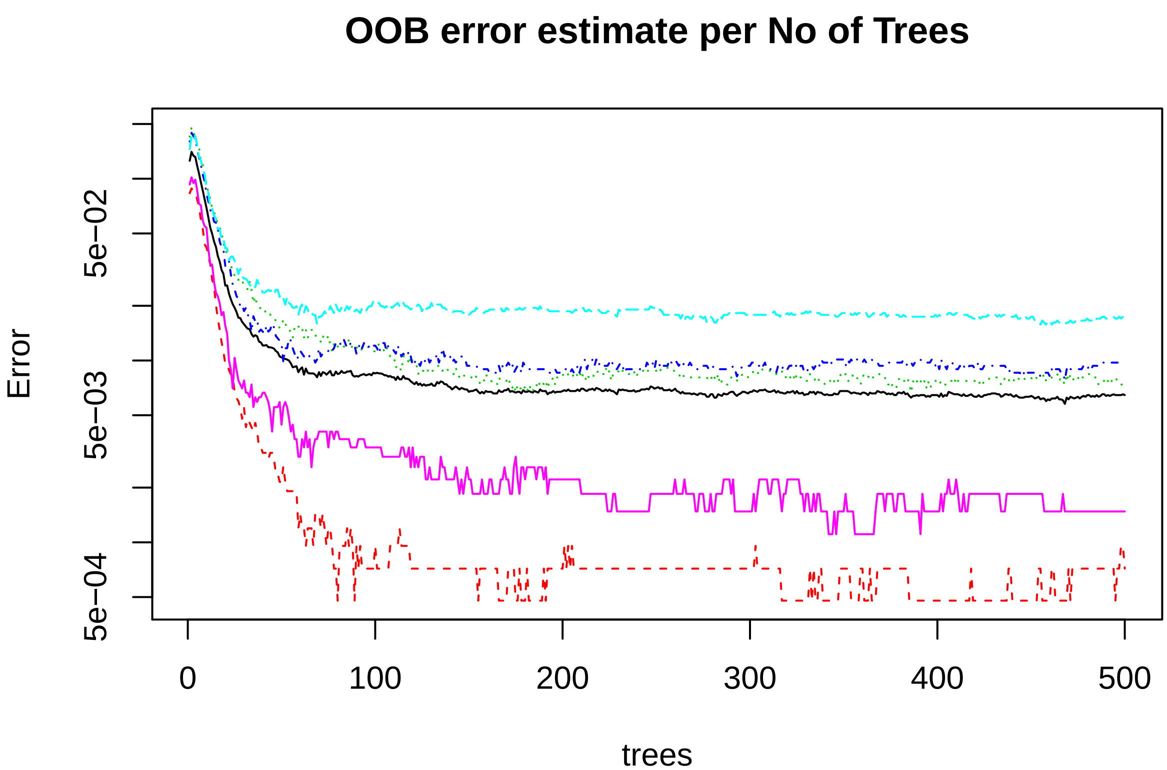
```
# Cross Validation using out of bag & Model Training
cross_validation_rf <- trainControl(method="oob",number=10,repeats=5,p=0.75)
model_rf <- suppressMessages(train(classe ~ ., method="rf", data=training, trControl=cross_validation_r

#Model Prediction
predict_rf <- predict(model_rf$finalModel,newdata=dev_test)

#Confusion Matrix
confusionMatrix(predict_rf,dev_test$classe)$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.9930669      0.9912296      0.9903250      0.9951940      0.2844617
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

```
# plot the Out of bag error estimates
plot(model_rf$finalModel,log="y", main ="OOB error estimate per No of Trees")
```



The accuracy of random forest models on the dev set is 99.29% on the dev set. In comparison with the accruacy of decision trees at 74.86%, the random forest models seems to perform good on the dev set.

Blind Test

The blind test of 20 testcases are to be evaluated with the best models and uploaded for grading. Hence we will use Random Forest models (which have the best perfomance overall).

```
#Random Forest
predict_rf_test <- predict(model_rf$finalModel,newdata=test)


write_prediction_to_file = function(x){
  n = length(x)
  for(i in 1:n){
    file_name = paste0("testcase_no_",i,".txt")
    write.table(x[i],file=file_name,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
write_prediction_to_file(predict_rf_test)
```

Conclusion

To conclude, for the 6 participant and 5 different action data, given accelerometer reading on the belt, forearm,arm and dumbell. We were able to predict the action with a high accuracy using random forests. Decison Trees did not provide as good a prediction in comparision to Random Trees for this task.

Bibliography

[1] Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. - Qualitative Activity Recognition of Weight Lifting Exercises: Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.