Practical Machine Learning Course Project

Mercia Silva

21 April 2016

Introduction to the Project

This project uses data from http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) has data collected from fitness devices located on the belt, forearm, arm, and dumbell of 6 participants who were asked to perform barbell lifts correctly and incorrectly in 5 different ways. The goal is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. Any of the other variables can be used in the prediction.

```
library(caret)
library(randomForest)
library(rpart)
library(rattle)
```

The Data

```
#download.file("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
", destfile = "pml-training.csv")
#download.file("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
, destfile = "pml-testing.csv")
train_data <- read.csv("pml-training.csv", sep = ",", na.strings = c("", "NA"))
result_data <- read.csv("pml-testing.csv", sep = ",", na.strings = c("", "NA"))
```

Having a look at the data with `head(train_data,3)` the data needs to be cleaned removing all NAs

```
features <- names(result_data[,colSums(is.na(result_data)) == 0])[8:59] #problem_id
not on train_data AND remove factors

# Only use features used in submit cases.
train_data <- train_data[,c(features,"classe")] # include classe
result_data <- result_data[,c(features,"problem_id")] # include problem_id
```

The result data is put aside and let's work only with the train data and dividing the train data into train (75%) and test (25%).

```
# bootstrap
set.seed(5)
inTrain = createDataPartition(train_data$classe, p = 0.75, list = F)
training = train_data[inTrain,]
testing = train_data[-inTrain,]
```

Using correlation to find the best features to use in the model

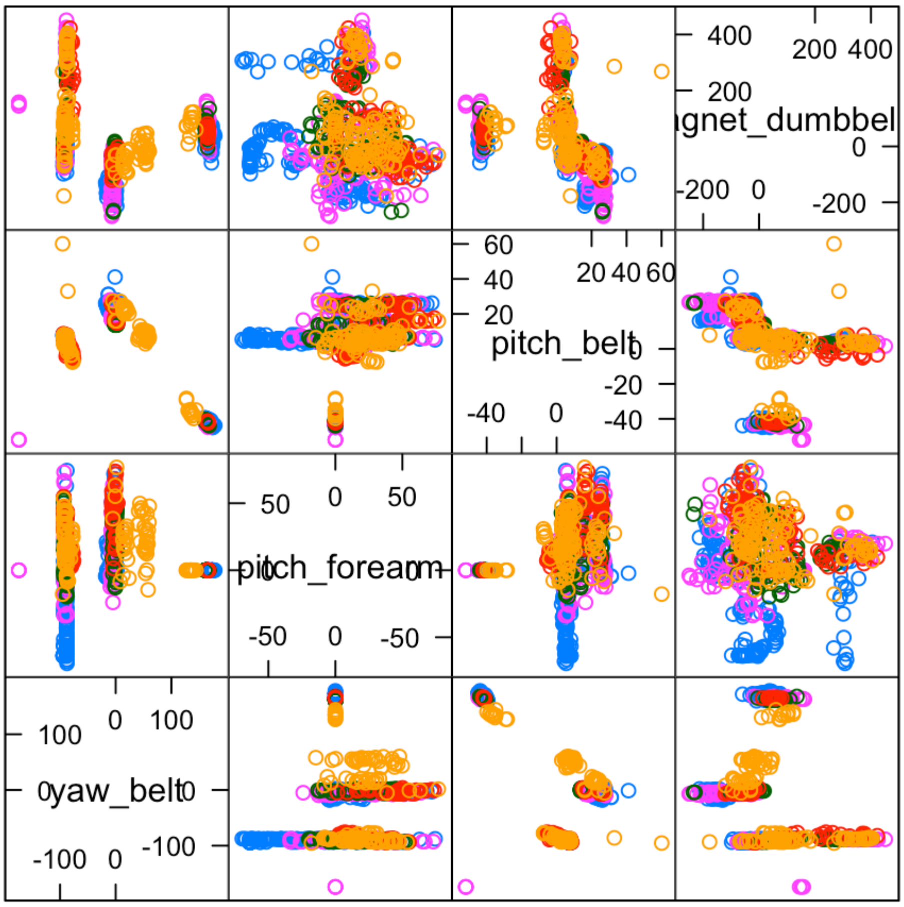
```
# feature selection
outcome = which(names(training) == "classe")
highCorrCols = findCorrelation(abs(cor(training[,-outcome])),0.90)
highCorrFeatures = names(training)[highCorrCols]
highCorrFeatures
```

```
## [1] "accel_belt_z"     "roll_belt"        "accel_belt_y"
## [4] "accel_belt_x"     "gyros_dumbbell_x" "gyros_dumbbell_z"
## [7] "gyros_arm_x"
```

```
training = training[,-highCorrCols]
outcome = which(names(training) == "classe")
```

Checking the relations of each feature and which are the most important ones to this model. The most important features are: pitch_belt, yaw_belt, total_accel_belt, gyros_belt_x

```
#feature importance
fsRF = randomForest(training[,-outcome], training[,outcome], importance = T)
rfImp = data.frame(fsRF$importance)
impFeatures = order(-rfImp$MeanDecreaseGini)
inImp = createDataPartition(train_data$classe, p = 0.05, list = F)
featurePlot(training[inImp,impFeatures[1:4]],training$classe[inImp], plot = "pairs"
)
```



Scatter Plot Matrix

Generating/training 3 models to compare: Decision Tree Model, K-Nearest Neighbors Model and Random Forest

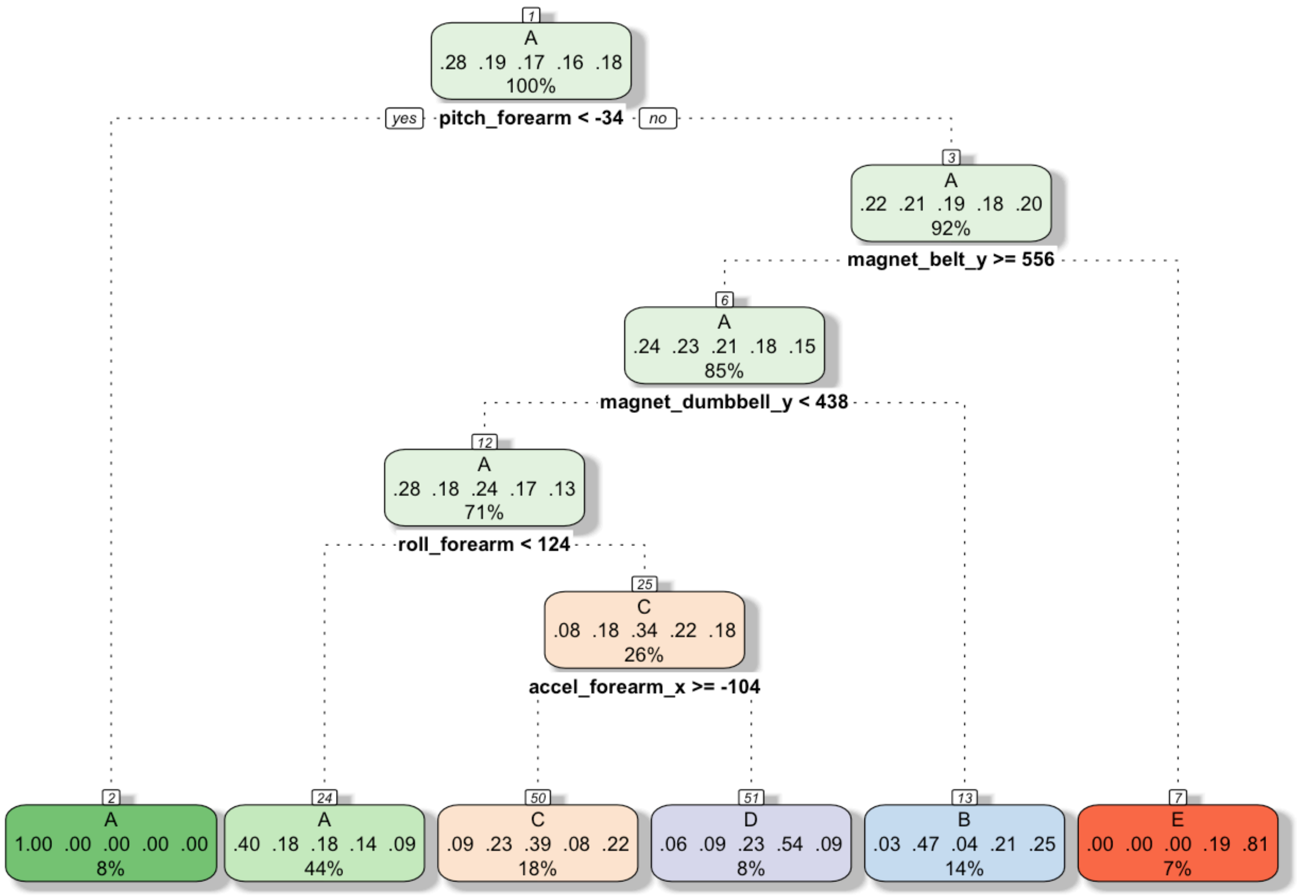
```
modelKNN = train(classe ~ ., training, method = "knn", trControl = trainControl(met
hod = "adaptive_cv"))
modelRF = train(classe ~ ., training, method = "rf", ntree = 200, trControl = train
Control(method = "oob"))
modelDT= train(classe ~ ., training, method="rpart")
resultsKNN = data.frame(modelKNN$results)
resultsRF = data.frame(modelRF$results)
resultsDT = data.frame(modelDT$results)
fitKNN = predict(modelKNN, testing)
fitRF = predict(modelRF, testing)
fitDT = predict(modelDT, testing)
```

Results for the Decision Tree Model:

```
confusionMatrix(fitDT, testing$classe)$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   4.951060e-01   3.398881e-01   4.810171e-01   5.092008e-01   2.844617e-01
## AccuracyPValue  McnemarPValue
##  7.446777e-212  9.980126e-322
```

```
fancyRpartPlot(modelDT$finalModel)
```



Rattle 2016-Apr-21 22:19:44 mercia

Results for the K-Nearest Neighbors Model:

```
confusionMatrix(fitKNN, testing$classe)$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   9.139478e-01   8.910976e-01   9.057482e-01   9.216506e-01   2.844617e-01
## AccuracyPValue  McnemarPValue
##   0.000000e+00   9.629415e-16
```

Results for the Random Forest Model:

```
confusionMatrix(fitRF, testing$classe)$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull 
##      0.9963295      0.9953570      0.9942053      0.9978232      0.2844617 
## AccuracyPValue  McnemarPValue 
##      0.0000000            NaN
```

Conclusion

Thee Random Forest model with Accuracy 0.9963295 is the best performing model.

Extra

Creating the data to submit to the project quiz:

```
submit = predict(modelRF, result_data)
answer = data.frame(submit)
answer
```

```
##    submit
## 1       B
## 2       A
## 3       B
## 4       A
## 5       A
## 6       E
## 7       D
## 8       B
## 9       A
## 10      A
## 11      B
## 12      C
## 13      B
## 14      A
## 15      E
## 16      E
## 17      A
## 18      B
## 19      B
## 20      B
```