Practical Machine Learning - Course Project

Augusto Hassel

Sunday, March 22, 2015

In this project, we will be using 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. The goal of this project is to predict the manner in which the individuals on our sample did the exercise.

First we will load our global options to work with.

```
library("caret");
library("ggplot2");library("knitr");library("tree");library("randomFor
est");library("gbm");library("e1071");

## Warning: package 'caret' was built under R version 3.1.2

## Loading required package: lattice
## Loading required package: ggplot2

## Warning: package 'tree' was built under R version 3.1.2
## Warning: package 'randomForest' was built under R version 3.1.2

## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.

## Warning: package 'gbm' was built under R version 3.1.2

## Loading required package: survival
## Loading required package: splines
##
## Attaching package: 'survival'
##
## The following object is masked from 'package:caret':
##
##     cluster
##
## Loading required package: parallel
## Loaded gbm 2.1

## Warning: package 'e1071' was built under R version 3.1.2

opts_chunk$set(echo=TRUE, results='asis', cache=TRUE)
```

We are going to load the data.

```
setwd("~/Cursos online/Data Science - Johns Hopkins/8 Machine
Learning/courseProject")
data <- read.csv(file = "pml-training.csv", header = T, na.strings =
c("", "NA", "NULL", "#DIV/0!"))
cases <- read.csv(file = "pml-testing.csv", header = T, na.strings =
c("", "NA", "NULL"))
```

After doing that, we are going to pre process de data, taking out variables that have NA values.

```
counting.na <- as.data.frame(apply(data, MARGIN = 2, function(x)
sum(is.na(x))))
names(counting.na) <- "count"
not.na <- row.names(counting.na)[which(counting.na$count == 0)]

data <- data[names(data) %in% not.na]
cases <- cases[names(cases) %in% not.na]
```

We divide the set into a training and a test subset to perform our analysis.

```
inTrain <- createDataPartition(y = data$classe, p = 0.7, list = F)
training <- data[inTrain, ]
testing <- data[-inTrain, ]
```

We came to realise that we could reduce even more our set by analysing the zero variance of the variables on the set. This variables won't help us to explain the model so the must be taken away.

```
nzv <- nearZeroVar(training)

training <- training[-nzv]
testing <- testing[-nzv]
cases <- cases[-nzv]
data <- data[-nzv]
```

We can take away time related and user related variables from the set.

```
toDelete <- c("X", "user_name", "new_window", "num_window",
"raw_timestamp_part_1", "raw_timestamp_part_2", "cvtd_timestamp")

training <- training[!(names(training) %in% toDelete)]
testing <- testing[!(names(testing) %in% toDelete)]
data <- data[!(names(data) %in% toDelete)]
cases <- cases[!(names(cases) %in% toDelete)]
```

Now that we have finally completed our pre proces, we can start modeling the data.

We are going to select one of the following models that have the greater accuracy on the testing subset:

1- Classification Tree

2- Prunning Tree

3- Random Forest

Classification Tree

```
treeModel <- tree(classe ~ ., data = training)
summary(treeModel)
```

```
## 
## Classification tree:
## tree(formula = classe ~ ., data = training)
## Variables actually used in tree construction:
##  [1] "roll_belt"            "pitch_forearm"        "roll_forearm"        
##  [4] "magnet_dumbbell_x"    "magnet_dumbbell_y"   
"magnet_dumbbell_z"   
##  [7] "roll_arm"             "yaw_belt"            
"total_accel_dumbbell"
## [10] "accel_forearm_x"      "accel_dumbbell_y"    
## Number of terminal nodes:  16 
## Residual mean deviance:  1.79 = 24600 / 13700 
## Misclassification error rate: 0.367 = 5047 / 13737
```

```
treePredClass <- predict(object = treeModel, newdata = testing, type = 
"class")
table(predictedValues = treePredClass, testing$classe)
```

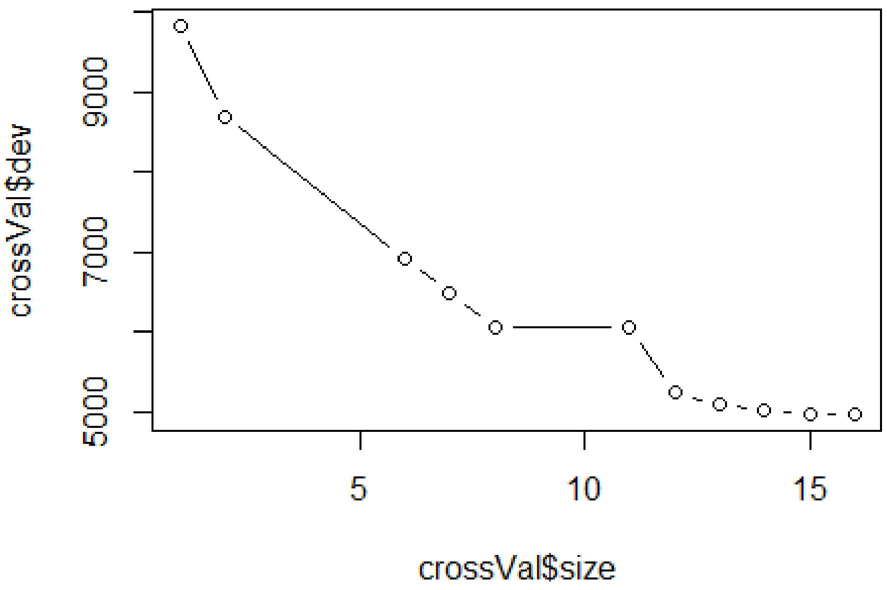
```
##                
## predictedValues    A    B    C    D    E
##               A 1297  220  103   16   30
##               B   54  290   13   15   11
##               C   76   90  689  152  100
##               D  240  364  153  759  270
##               E    7  175   68   22  671
```

```
treeA <- mean(treePredClass == testing$classe)
```

We have an accuracy of 0.6297 on our Classification Tree. Let us move on into other models.

Prunning Tree

```
crossVal <- cv.tree(object = treeModel, FUN = prune.misclass)
```



"Dev" give us the cross validation error.

Now we select the maximum nodes for our model.

```
pruneTree <- prune.misclass(tree = treeModel, best = 16)
```

With 16 nodes the miscclasification is being reduced consirably. Now we evaluate the accuracy.

```
prunePredClass <- predict(object = pruneTree, newdata = testing, type
= "class")
table(prunePredClass, testing$classe)

##
## prunePredClass    A    B    C    D    E
##              A 1297  220  103   16   30
##              B   54  290   13   15   11
##              C   76   90  689  152  100
##              D  240  364  153  759  270
##              E    7  175   68   22  671

pruneA <- mean(prunePredClass == testing$classe)
```

We have an accuracy of 0.6297 on our Prunne Tree. Let us move on into other models because this is to similar to our classification tree.

Random Forest

```
randomForesModel <- randomForest(classe ~ . , data = training,
importance = T)
randomForestPredict <- predict(object = randomForesModel, newdata =
testing)
```

```
table(randomForestPredict, testing$classe)

##
## randomForestPredict    A    B    C    D    E
##                   A 1674    6    0    0    0
##                   B    0 1125    4    0    0
##                   C    0    8 1022    9    1
##                   D    0    0    0  952    1
##                   E    0    0    0    3 1080

forestA <- mean(randomForestPredict == testing$classe)
```

We have an accuracy of 0.9946 on our Random Forest. This has been the better accuracy so far.

Conclusion

We are drawn to select the Random Forest model because of it accuracy and performance over the other models. We have tested ourside this work SVM and Boosting models, but the accuracy hasn't improved that much in terms of the performance, that's why we keep this models as our selected one for this case.

Accuracy between models.

1- Classification Tree: 0.6297

2 - Prunne Tree: 0.6297

3 - Random Forest: 0.9946

Cases results

```
CasePredictions <- predict(object = randomForesModel, newdata = cases)
CasePredictions

##  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
```