Practical Machine Learning

Final Project

Background

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. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. 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, your goal will be to use 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. 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).

Introduction

This is the final project of the practical machine learning course conducted by John Hopkins University. The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. The total number of participants are six. Further we will be using a Prediction model to predict 20 different test cases. The software used for the analysis is R version 3.3.0.

Data

The data for the project was taken from http://groupware.les.inf.puc-rio.br/har. More information on the same can be found here. The training data for this project are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv and The test data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv.

For the analysis, we are splitting the data into Training set and Testing set in the ratio 60:40. The training set consists of 11776 observations and 160 variables and the total number of observations in testing test are 7846 and 160 variables.

To get better results we are cleaning the data before the main analysis. For this purpose, we removed all the variables with the near zero variances, removed the ID column and removed the variables with too many missing values (We removed the variables which have more than 60% missing values). The same was done for the test set too. In the end we were left with 11776 observations and 58 variables in the training set and 20 observations and 57 variables in the test data.

Data Modelling

We will use the random forest ML algorithm to fit a predictive model for activity recognition and to check its behavior. The Random forest automatically chooses the important the variables. A five-fold cross validation is used in the algorithm.

```
Call:
randomForest(x = x, y = y, mtry = param$mtry)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 27

        OOB estimate of  error rate: 0.28%
Confusion matrix:
     A    B    C    D      E     class.error
A  3905  1    0    0      0     0.0002560164
B   3   2650  4    1      0     0.0030097818
C   0    9   2386  1      0     0.0041736227
D   0    0    11  2240    1     0.0053285968
E   0    2    0    6     2517   0.0031683168
```

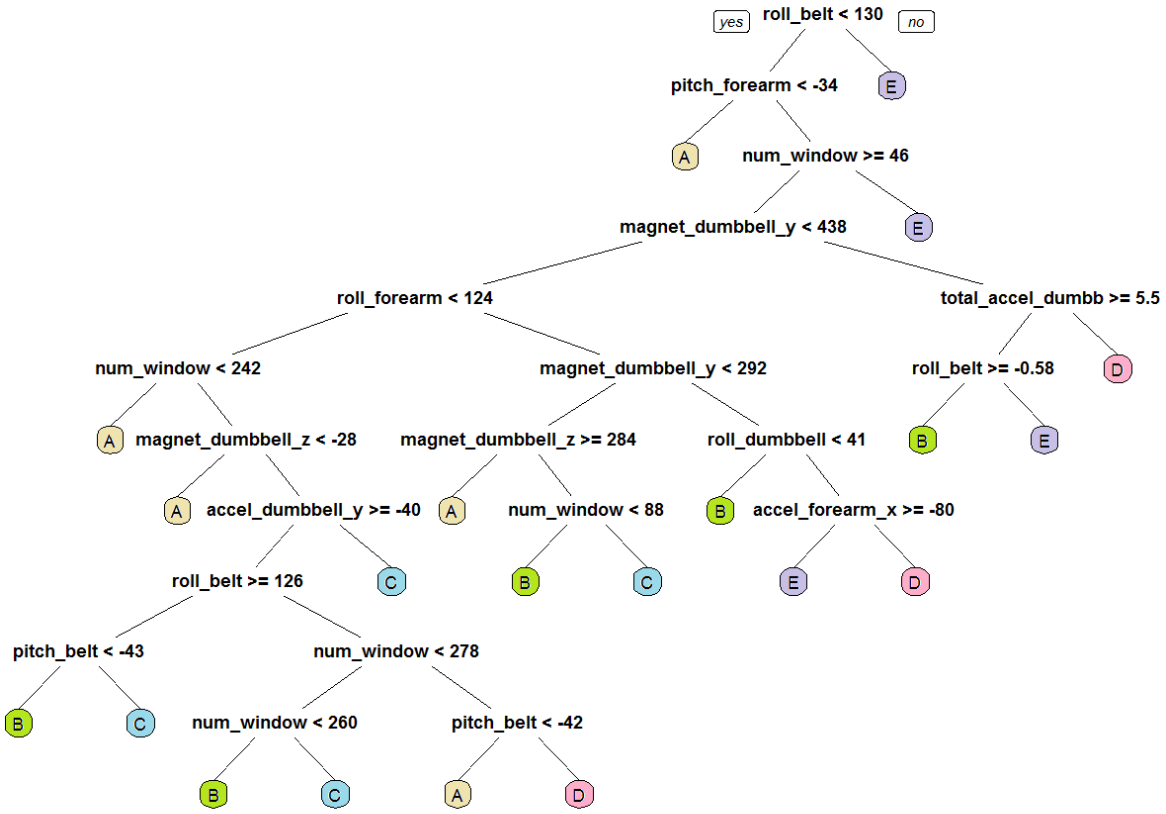
From the above results, we can conclude that the number tree are 500 and the number of variables tried at each split are 27. The same will be done for the test data. The results are shown below:

```
Confusion Matrix and Statistics


          Reference
Prediction    A    B    C    D    E
         A   1673  0    0    0    1
         B    2   1137  0    0    0
         C    0    1   1025  0    0
         D    0    0    9   955   0
         E    0    0    0    0   1082
```

```
Overall Statistics

               Accuracy : 0.9978
                 95% CI : (0.9962, 0.9988)
    No Information Rate : 0.2846
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9972
Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9988   0.9991   0.9913   1.0000   0.9991
Specificity            0.9998   0.9996   0.9998   0.9982   1.0000
Pos Pred Value         0.9994   0.9982   0.9990   0.9907   1.0000
Neg Pred Value         0.9995   0.9998   0.9981   1.0000   0.9998
Prevalence             0.2846   0.1934   0.1757   0.1623   0.1840
Detection Rate         0.2843   0.1932   0.1742   0.1623   0.1839
Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
Balanced Accuracy      0.9993   0.9993   0.9955   0.9991   0.9995
```

The accuracy for this model is found to be 99.78 and the estimated out-of-sample error is 0.28% which can be seen in the results. The error is less the 1% which shows that the model is well constructed.

Decision Tree



Prediction for the 20 different test cases:

The final part of this project is to use prediction model to predict the 20 test cases. The results are as follows:

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

Appendix

R Code:

```
library(caret)

library(rpart)

library(rpart.plot)

library(RColorBrewer)

library(randomForest)

set.seed(12345)

trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"

testUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

training <- read.csv(url(trainUrl), na.strings=c("NA","#DIV/0!",""))

testing <- read.csv(url(testUrl), na.strings=c("NA","#DIV/0!",""))

inTrain <- createDataPartition(y=training$classe, p=0.6, list=FALSE)

myTraining <- training[inTrain, ]; myTesting <- training[-inTrain, ]

dim(myTraining); dim(myTesting)

myDataNZV <- nearZeroVar(myTraining, saveMetrics=TRUE)

myNZVvars <- names(myTraining) %in% c("new_window", "kurtosis_roll_belt", "kurtosis_pi
cth_belt",

"kurtosis_yaw_belt", "skewness_roll_belt", "skewness_roll_belt.1", "skewness_yaw_belt"
,

"max_yaw_belt", "min_yaw_belt", "amplitude_yaw_belt", "avg_roll_arm", "stddev_roll_arm
",

"var_roll_arm", "avg_pitch_arm", "stddev_pitch_arm", "var_pitch_arm", "avg_yaw_arm",

"stddev_yaw_arm", "var_yaw_arm", "kurtosis_roll_arm", "kurtosis_picth_arm",

"kurtosis_yaw_arm", "skewness_roll_arm", "skewness_pitch_arm", "skewness_yaw_arm",

"max_roll_arm", "min_roll_arm", "min_pitch_arm", "amplitude_roll_arm", "amplitude_pitc
h_arm",

"kurtosis_roll_dumbbell", "kurtosis_picth_dumbbell", "kurtosis_yaw_dumbbell", "skewnes
s_roll_dumbbell",

"skewness_pitch_dumbbell", "skewness_yaw_dumbbell", "max_yaw_dumbbell", "min_yaw_dumbb
ell",
```

```
"amplitude_yaw_dumbbell", "kurtosis_roll_forearm", "kurtosis_picth_forearm", "kurtosis
_yaw_forearm",

"skewness_roll_forearm", "skewness_pitch_forearm", "skewness_yaw_forearm", "max_roll_f
orearm",

"max_yaw_forearm", "min_roll_forearm", "min_yaw_forearm", "amplitude_roll_forearm",

"amplitude_yaw_forearm", "avg_roll_forearm", "stddev_roll_forearm", "var_roll_forearm"
,

"avg_pitch_forearm", "stddev_pitch_forearm", "var_pitch_forearm", "avg_yaw_forearm",

"stddev_yaw_forearm", "var_yaw_forearm")

myTraining <- myTraining[!myNZVvars]

dim(myTraining)

myTraining <- myTraining[c(-1)]

trainingV3 <- myTraining #creating another subset to iterate in loop

for(i in 1:length(myTraining)) { #for every column in the training dataset

        if( sum( is.na( myTraining[, i] ) ) /nrow(myTraining) >= .6 ) { #if n?? NAs >
60% of total observations

        for(j in 1:length(trainingV3)) {

            if( length( grep(names(myTraining[i]), names(trainingV3)[j]) ) ==1)  { #if
the columns are the same:

                trainingV3 <- trainingV3[ , -j] #Remove that column

            }

        }

    }

}

dim(trainingV3)

myTraining <- trainingV3

rm(trainingV3)

clean1 <- colnames(myTraining)

clean2 <- colnames(myTraining[, -58])

myTesting <- myTesting[clean1]

testing <- testing[clean2]
```

```
trainFit <- trainControl(method="cv", number=5, verboseIter=F)
modelFit <- train(classe ~ ., data=trainSet, method="rf", trControl=trainFit)
modelFit$finalModel
preds <- predict(modelFit, newdata=testSet)
confusionMatrix(testSet$classe, preds)
tree <- rpart(classe ~ ., data=trainSet, method="class")
prp(tree)
predictionsB2 <- predict(modFitB1, testing, type = "class")
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(predictionsB2)
```