Machine Learning 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 dumbbell 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).

Data

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. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

What you should submit

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. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

1. Your submission should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-).
2. You should also apply your machine learning algorithm to the 20 test cases available in the test data above. Please submit your predictions in appropriate format to the programming assignment for automated grading. See the programming assignment for additional details.

Reproduceability

The code should be run with the random seed 1231. Necessary packages for the model are listed below.

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.2.3
```

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

```
library(rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 3.2.3
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.2.3
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```
set.seed(1231)
```

Loading data

Load Training and Test data sets from web. Note: if files were previously saved in the relevant project, commented code can be used to load the data faster.

```
TrainingLocation <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
TrainingData <- read.csv(url(TrainingLocation), na.strings=c("NA","#DIV/0!",""))

TestingLocation <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
TestingData <- read.csv(url(TestingLocation), na.strings=c("NA","#DIV/0!",""))

##To use this if files are saved in the project (faster variant)

#TrainingData <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!",""))
#TestingData <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!",""))
```

Cleaning data

```
# 1. Remove columns that containes more then > 95% of  NAs
TrainingData<-TrainingData[ , ! apply(TrainingData , 2 , function(x)
        sum(is.na(x))/nrow(TrainingData) >0.95)]

# 2. Remove first seven columns that are not relevant for the model (user names, IDs, window).
TrainingData <- subset(TrainingData, select = -c(1:7) )
dim(TrainingData)
```

```
## [1] 19622    53
```

The number of variables reduced from 160 to 53.

Training data set split for cross validation

Data split - 60% of training set data to built a model, and 40% to test the model

```
train <- createDataPartition(y=TrainingData$classe,p=.60,list=FALSE)
SubTraining <- TrainingData[train,]
SubTesting <- TrainingData[-train,]
dim(SubTraining); dim(SubTesting)
```

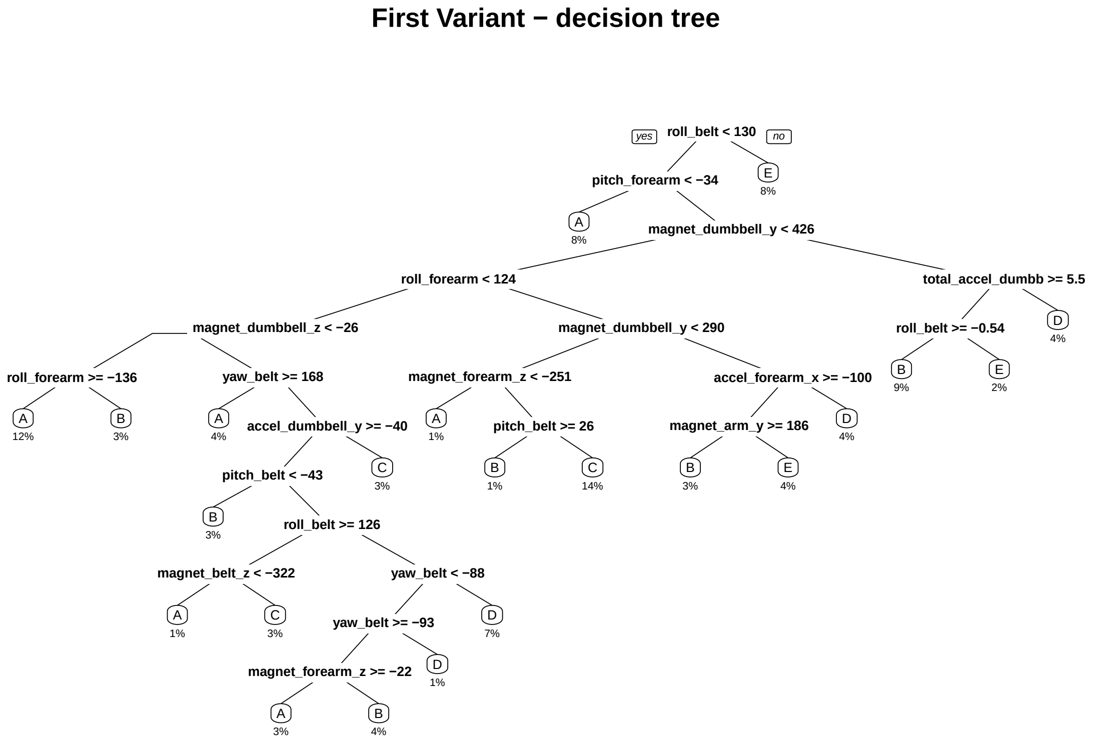
```
## [1] 11776    53
```

```
## [1] 7846   53
```

First variant - Decision Tree

```
# Grow the tree
FirstModel <- rpart(classe ~ ., data=SubTraining, method="class")

# Plot of the Decision Tree
rpart.plot(FirstModel, main="First Variant - decision tree", extra=100, under=TRUE, faclen=0)
```



```
# Predict based on the decision tree
FirstModelPred <- predict(FirstModel, SubTesting, type = "class")

# Test the result using testing subset
confusionMatrix(FirstModelPred, SubTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1943  238   37   63   36
##          B  107 1010  199  140  174
##          C   49  109 1018  180  159
##          D   97  113   81  839  132
##          E   36   48   33   64  941
##
## Overall Statistics
##
##                Accuracy : 0.733
##                  95% CI : (0.723, 0.7427)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6618
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8705   0.6653   0.7442   0.6524   0.6526
## Specificity            0.9334   0.9020   0.9233   0.9355   0.9717
## Pos Pred Value         0.8386   0.6196   0.6719   0.6648   0.8387
## Neg Pred Value         0.9477   0.9183   0.9447   0.9321   0.9255
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2476   0.1287   0.1297   0.1069   0.1199
## Detection Prevalence   0.2953   0.2077   0.1931   0.1608   0.1430
## Balanced Accuracy      0.9020   0.7837   0.8337   0.7940   0.8122
```

Confusion matrix shows that the first model - decision tree can predict classes with accuracy 0.733, confidence interval 95%: 0.723,07427. Due to the fact that there are quite a lot of variables the random forest variant could be a better fit.

Second variant - Random forest

```r
# Build random forest model
SecondModel <- randomForest(classe ~. , data=SubTraining, method="class")
print(SecondModel)
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = SubTraining, method = "class")
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.7%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3343    4    0    0    1 0.001493429
## B   11 2261    7    0    0 0.007898201
## C    0   19 2032    3    0 0.010710808
## D    0    0   27 1900    3 0.015544041
## E    0    0    3    5 2157 0.003695150
```

```
# Predict based on the SecondModel
SecondModelPred <- predict(SecondModel, SubTesting, type = "class")

# Test the result using testing subset
confusionMatrix(SecondModelPred, SubTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    9    0    0    0
##          B    0 1500    7    0    0
##          C    0    9 1360   11    1
##          D    0    0    1 1274    4
##          E    0    0    0    1 1437
##
## Overall Statistics
##
##                Accuracy : 0.9945
##                  95% CI : (0.9926, 0.996)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9931
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9881   0.9942   0.9907   0.9965
## Specificity            0.9984   0.9989   0.9968   0.9992   0.9998
## Pos Pred Value         0.9960   0.9954   0.9848   0.9961   0.9993
## Neg Pred Value         1.0000   0.9972   0.9988   0.9982   0.9992
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1912   0.1733   0.1624   0.1832
## Detection Prevalence   0.2856   0.1921   0.1760   0.1630   0.1833
## Balanced Accuracy      0.9992   0.9935   0.9955   0.9950   0.9982
```

Reasone for decision

The second model build with random forest can predict with accuracy 0.9944, confidence interval 95%: 0.9925, 09959. Random forest model achieves much better accuracy in comparison with the decision tree model tested above.

Expected out of sample error

When we use random forest model the expected out-of-sample error is around 0.5% and is calculated as 1 minus accuracy for predictions made against the cross-validation set.

Cross-validation

To perform a Cross-validation the training data set split into 2 sets: SubTraining (60%) and SubTesting (40%). The model is fitted in SubTraining data set and tested SubTesting data set. The random forest model gave an accuracy above 99% on cross-validation data (SubTesting data set). It means that in Testing Data set out of 20 test cases only very few or none of the cases will be wrong.

Predict 20 different test cases

```r
# 1. Remove columns that containes more then > 95% of  NAs
TestingData<-TestingData[ , ! apply(TestingData , 2 , function(x)
      sum(is.na(x))/nrow(TestingData) >0.95)]
# 2. Remove first seven columns that are not relevant for the model (user names, IDs, window).
TestingData <- subset(TestingData, select = -c(1:7) )
dim(TestingData)
```

```
## [1] 20 53
```

```r
# Predition with the second model - random forest
PredictTest <- predict(SecondModel, TestingData, type="class")
PredictTest
```

```
##  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
```

Submission

```r
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(PredictTest)
```