CorrectMovementPrediction

The project goal will be to predict a correct exercise movement using data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. Participants performed one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different ways: "correctly" (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D), throwing the hips to the front (Class E).

Given these data try to predict what class of movement in 20 cases.

Obtaining data...

```
trainUrl <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
testUrl <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'
# a first look at the data shows different kinds of NA elements. Change to one.
training <- read.csv(url(trainUrl), na.strings=c("NA","#DIV/0!",""))
testing <- read.csv(url(testUrl), na.strings=c("NA","#DIV/0!",""))
dim(training); dim(testing)
```

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

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

Need to prepare data for modeling

```
library(caret)
```

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

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

```
library(rpart)
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
# remove variables that will not affect classe. Assuming time stamp does not have effect on classe.
trainSet <- training[,-(1:7)]
testSet <- testing[,-(1:7)]

# trainSet has "classe" while testSet has "problem_id"
# class(trainSet$classe); class(testSet$problem_id)
# [1] "factor"
# [1] "integer"
```

```
testSet$problem_id <- as.factor(testSet$problem_id)

# remove variables with no variation as they are not likely to affect change in classe
varList <- nearZeroVar(trainSet)
trainSet <- trainSet[,-varList]
testSet <- testSet[,-varList]

# remove variables with high NA content
NAset <- sapply(trainSet, function(x) mean(is.na(x))) > 0.70
trainSet <- trainSet[, NAset == FALSE]
NAset <- sapply(testSet, function(x) mean(is.na(x))) > 0.70
testSet <- testSet[, NAset == FALSE]

# split training set into training and validation set
set.seed(123)
inTrain <- createDataPartition(y=trainSet$classe, p=0.6, list=FALSE)
trainSettrain <- trainSet[inTrain, ]; validation <- trainSet[-inTrain, ]
dim(trainSettrain); dim(validation)
```

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

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

Trying recursive partitioning and regression tree. The "classe" variable in the training set provides the movement class to train on.

```
rpartModel <- rpart(classe ~ ., data = trainSettrain, method="class")
rpartpredict <- predict(rpartModel, newdata = validation, type = "class")
confusionMatrix(rpartpredict, validation$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2059  203   18  128   79
##          B   60 1086  211   70   93
##          C   20   83  998  184   83
##          D   71  138   81  863  120
##          E   22    8   60   41 1067
##
## Overall Statistics
##
##                Accuracy : 0.774
##                  95% CI : (0.7646, 0.7832)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.713
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9225   0.7154   0.7295   0.6711   0.7399
## Specificity            0.9238   0.9314   0.9429   0.9375   0.9795
```

```
## Pos Pred Value         0.8279   0.7145   0.7295   0.6779   0.8907
## Neg Pred Value         0.9677   0.9317   0.9429   0.9356   0.9436
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2624   0.1384   0.1272   0.1100   0.1360
## Detection Prevalence   0.3170   0.1937   0.1744   0.1622   0.1527
## Balanced Accuracy      0.9231   0.8234   0.8362   0.8043   0.8597
```

Accuracy at 0.774

Trying the random forest method

```
rfModel <- randomForest(classe ~. , data= trainSettrain)
rfpredict <- predict(rfModel, newdata = validation, type = "class")
confusionMatrix(rfpredict, validation$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2229    7    0    0    0
##          B    2 1508   11    0    0
##          C    0    3 1354   11    4
##          D    0    0    3 1275    3
##          E    1    0    0    0 1435
##
## Overall Statistics
##
##                Accuracy : 0.9943
##                  95% CI : (0.9923, 0.9958)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9927
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9987   0.9934   0.9898   0.9914   0.9951
## Specificity            0.9988   0.9979   0.9972   0.9991   0.9998
## Pos Pred Value         0.9969   0.9915   0.9869   0.9953   0.9993
## Neg Pred Value         0.9995   0.9984   0.9978   0.9983   0.9989
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2841   0.1922   0.1726   0.1625   0.1829
## Detection Prevalence   0.2850   0.1939   0.1749   0.1633   0.1830
## Balanced Accuracy      0.9987   0.9957   0.9935   0.9953   0.9975
```

Accuracy at 0.993 giving an out of sample error of 0.007

As random forest has higher accuracy let's go with it for predicting the test set

```
rfTestpredict <- predict(rfModel, newdata = testSet, type = "class")
rfTestpredict
```

```
##  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
```

Out of curiosty, how different does the rpartModel predict?

```
rpartTestpredict <- predict(rpartModel, newdata = testSet, type = "class")
rpartTestpredict
```

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

Indeed, there is a difference in just 20 samples.