Practical Machine Learning Final Project

# loading libraries  
library(pbkrtest)

## Loading required package: lme4

## Loading required package: Matrix

library(car)  
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(data.table)

## Assigning the datasets and printing out their dimensions which turn out to be contain 19622 observations with 160 variables. The testing data contains 20 observations and 160 variables.

ptrain <- read.csv("~/Desktop/Coursera/Practical Machine Learning/pml-training.csv")  
ptest <- read.csv("~/Desktop/Coursera/Practical Machine Learning/pml-testing.csv")  
dim(ptrain)

## [1] 19622 160

dim(ptest)

## [1] 20 160

## The outcome to predict in the training set is the variable classe. The next steps include removing any unwanted data, including NA's

sum(complete.cases(ptrain))

## [1] 406

ptrain <- ptrain[, colSums(is.na(ptrain)) == 0]   
ptest <- ptest[, colSums(is.na(ptest)) == 0]   
classe <- ptrain$classe  
ptrainremove <- grepl("^X|timestamp|window", names(ptrain))  
ptrain <- ptrain[, !ptrainremove]  
cleantrain <- ptrain[, sapply(ptrain, is.numeric)]  
cleantrain$classe <- classe  
ptestremove <- grepl("^X|timestamp|window", names(ptest))  
ptest <- ptest[, !ptestremove]  
cleantest <- ptest[, sapply(ptest, is.numeric)]  
dim(cleantrain)

## [1] 19622 53

dim(cleantest)

## [1] 20 53

## The dimensions of the now cleaned test and training data are 19622 obervations with 53 variables, and 20 observations with 53 variables, respectively. Our next step is to split the training data into the recommened %70 for training, %30 for testing

set.seed(99)  
settrain <- createDataPartition(cleantrain$classe, p=0.70, list=F)  
traindata <- cleantrain[settrain, ]  
testdata <- cleantrain[-settrain, ]  
traindata1 <-cleantrain[settrain[1]]

## We fit a predictive model choosing a 5-fold cross validation to the training data using the 'random forest' method

controlrandomforest <- trainControl(method="cv", 5)  
modelrandomforest <- train(classe ~ ., data=traindata, method="rf", trControl=controlrandomforest, ntree=250)

## Loading required package: 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

modelrandomforest

## Random Forest   
##   
## 13737 samples  
## 52 predictor  
## 5 classes: 'A', 'B', 'C', 'D', 'E'   
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold)   
## Summary of sample sizes: 10988, 10990, 10990, 10990, 10990   
## Resampling results across tuning parameters:  
##   
## mtry Accuracy Kappa   
## 2 0.9898812 0.9871984  
## 27 0.9913373 0.9890418  
## 52 0.9869695 0.9835160  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was mtry = 27.

## Then, we estimate the performance of the model on the validation data set.

predicting\_Rf <- predict(modelrandomforest, testdata)  
confMatRandForest <- confusionMatrix(testdata$classe, predicting\_Rf)  
confMatRandForest

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1671 1 1 0 1  
## B 14 1122 3 0 0  
## C 0 2 1019 5 0  
## D 0 1 11 951 1  
## E 0 0 6 7 1069  
##   
## Overall Statistics  
##   
## Accuracy : 0.991   
## 95% CI : (0.9882, 0.9932)  
## No Information Rate : 0.2863   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9886   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9917 0.9964 0.9798 0.9875 0.9981  
## Specificity 0.9993 0.9964 0.9986 0.9974 0.9973  
## Pos Pred Value 0.9982 0.9851 0.9932 0.9865 0.9880  
## Neg Pred Value 0.9967 0.9992 0.9957 0.9976 0.9996  
## Prevalence 0.2863 0.1913 0.1767 0.1636 0.1820  
## Detection Rate 0.2839 0.1907 0.1732 0.1616 0.1816  
## Detection Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Balanced Accuracy 0.9955 0.9964 0.9892 0.9924 0.9977

accuracy <- postResample(predicting\_Rf, testdata$classe)  
accuracy

## Accuracy Kappa   
## 0.9909941 0.9886066

anw <- 1 - as.numeric(confusionMatrix(testdata$classe, predicting\_Rf)$overall[1])  
anw

## [1] 0.009005947

## In conclusion, the accuracy of the model based on this test data was estimated at 99.1% and the out-of-sample error was estimated at 0.57%.

## Moving on to testing the model on the dataset provided

result <- predict(modelrandomforest, cleantest[,-length(names(cleantest))])  
result

## [1] 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

## plotting the results of the matrix

plot(confMatRandForest$table, col = confMatRandForest$byClass,   
 main = paste("Random Forest - Accuracy =",  
 round(confMatRandForest$overall['Accuracy'], 4)))

