Final\_Project-Practical Machine Learning

library(caret); library(rattle); library(rpart); library(rpart.plot)

## Loading required package: lattice

## Loading required package: ggplot2

## Rattle: A free graphical interface for data mining with R.  
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

library(randomForest); library(repmis)

## 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

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(training$classe, p=0.6, list=FALSE)  
inTrain <- createDataPartition(training$classe, p=0.6, list=FALSE)  
myTraining <- training[inTrain, ]  
myTesting <- training[-inTrain, ]  
dim(myTraining); dim(myTesting)

## [1] 11776 160

## [1] 7846 160

#Cleaning the data  
nzv <- nearZeroVar(myTraining, saveMetrics=TRUE)  
myTraining <- myTraining[,nzv$nzv==FALSE]  
nzv<- nearZeroVar(myTesting,saveMetrics=TRUE)  
myTesting <- myTesting[,nzv$nzv==FALSE]  
myTraining <- myTraining[c(-1)]  
trainingV3 <- myTraining  
for(i in 1:length(myTraining)) {if( sum( is.na( myTraining[, i] ) ) /nrow(myTraining) >= .7) {for(j in 1:length(trainingV3)) {if( length( grep(names(myTraining[i]), names(trainingV3)[j]) ) == 1) { trainingV3 <- trainingV3[ , -j]}}}}  
#Clean variables with more than 60% NA  
myTraining <- trainingV3  
rm(trainingV3)  
clean1 <- colnames(myTraining)  
clean2 <- colnames(myTraining[, -58])   
myTesting <- myTesting[clean1]  
testing <- testing[clean2]   
dim(myTesting)

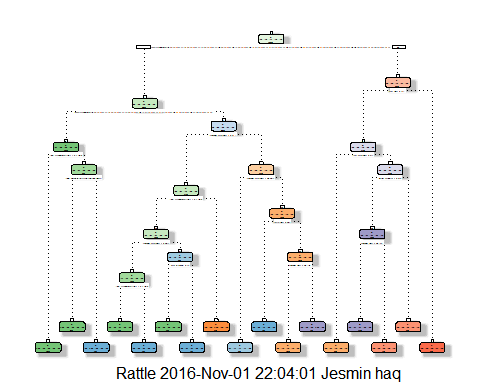
## [1] 7846 58

dim(testing)

## [1] 20 57

for (i in 1:length(testing) ) {for(j in 1:length(myTraining)) {  
if( length( grep(names(myTraining[i]), names(testing)[j]) ) == 1) { class(testing[j]) <- class(myTraining[i])}}}  
testing <- rbind(myTraining[2, -58] , testing)  
testing <- testing[-1,]

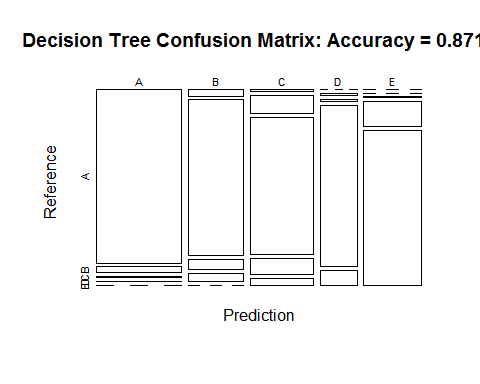
#Prediction with Decision Trees  
set.seed(12345)  
modFitA1 <- rpart(classe ~ ., data=myTraining, method="class")  
fancyRpartPlot(modFitA1)



predictionsA1 <- predict(modFitA1, myTesting, type = "class")  
cmtree <- confusionMatrix(predictionsA1, myTesting$classe)  
cmtree

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 2157 75 5 1 0  
## B 56 1260 77 65 0  
## C 19 170 1265 150 61  
## D 0 13 14 859 81  
## E 0 0 7 211 1300  
##   
## Overall Statistics  
##   
## Accuracy : 0.8719   
## 95% CI : (0.8643, 0.8792)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.8379   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9664 0.8300 0.9247 0.6680 0.9015  
## Specificity 0.9856 0.9687 0.9383 0.9835 0.9660  
## Pos Pred Value 0.9638 0.8642 0.7598 0.8883 0.8564  
## Neg Pred Value 0.9866 0.9596 0.9833 0.9379 0.9776  
## Prevalence 0.2845 0.1935 0.1744 0.1639 0.1838  
## Detection Rate 0.2749 0.1606 0.1612 0.1095 0.1657  
## Detection Prevalence 0.2852 0.1858 0.2122 0.1232 0.1935  
## Balanced Accuracy 0.9760 0.8994 0.9315 0.8257 0.9337

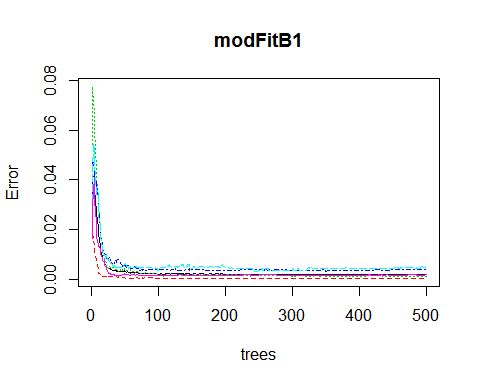
plot(cmtree$table, col = cmtree$byClass, main = paste("Decision Tree Confusion Matrix: Accuracy =", round(cmtree$overall['Accuracy'], 4)))



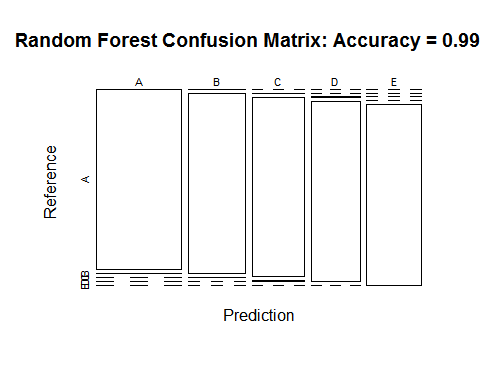
#Prediction with Random Forests  
set.seed(12345)  
modFitB1 <- randomForest(classe ~ ., data=myTraining)  
predictionB1 <- predict(modFitB1, myTesting, type = "class")  
cmrf <- confusionMatrix(predictionB1, myTesting$classe)  
cmrf

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 2231 3 0 0 0  
## B 1 1514 1 0 0  
## C 0 1 1366 6 0  
## D 0 0 1 1280 0  
## E 0 0 0 0 1442  
##   
## Overall Statistics  
##   
## Accuracy : 0.9983   
## 95% CI : (0.9972, 0.9991)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9979   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9996 0.9974 0.9985 0.9953 1.0000  
## Specificity 0.9995 0.9997 0.9989 0.9998 1.0000  
## Pos Pred Value 0.9987 0.9987 0.9949 0.9992 1.0000  
## Neg Pred Value 0.9998 0.9994 0.9997 0.9991 1.0000  
## Prevalence 0.2845 0.1935 0.1744 0.1639 0.1838  
## Detection Rate 0.2843 0.1930 0.1741 0.1631 0.1838  
## Detection Prevalence 0.2847 0.1932 0.1750 0.1633 0.1838  
## Balanced Accuracy 0.9995 0.9985 0.9987 0.9976 1.0000

plot(modFitB1)



plot(cmrf$table, col = cmtree$byClass, main = paste("Random Forest Confusion Matrix: Accuracy =", round(cmrf$overall['Accuracy'], 4)))



#Prediction with Generalized Boosted Regression  
set.seed(12345)  
fitControl <- trainControl(method = "repeatedcv",number = 5, repeats = 1)  
gbmFit1 <- train(classe ~ ., data=myTraining, method = "gbm",trControl = fitControl, verbose = FALSE)

## Loading required package: gbm

## Loading required package: survival

##   
## Attaching package: 'survival'

## The following object is masked from 'package:caret':  
##   
## cluster

## Loading required package: splines

## Loading required package: parallel

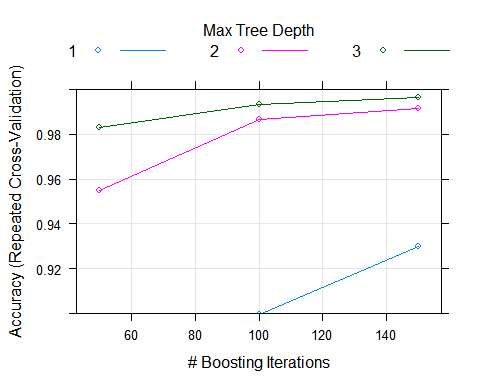
## Loaded gbm 2.1.1

## Loading required package: plyr

fitControl <- trainControl(method = "repeatedcv",number=5, repeats=1)  
gbmFit1 <- train(classe ~ ., data=myTraining, method = "gbm",trControl = fitControl,verbose = FALSE)  
gbmFinMod1 <- gbmFit1$finalModel  
gbmPredTest <- predict(gbmFit1, newdata=myTesting)  
gbmAccuracyTest <- confusionMatrix(gbmPredTest, myTesting$classe)  
gbmAccuracyTest

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 2231 8 0 0 0  
## B 1 1503 0 0 0  
## C 0 4 1357 7 0  
## D 0 3 11 1274 3  
## E 0 0 0 5 1439  
##   
## Overall Statistics  
##   
## Accuracy : 0.9946   
## 95% CI : (0.9928, 0.9961)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9932   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9996 0.9901 0.9920 0.9907 0.9979  
## Specificity 0.9986 0.9998 0.9983 0.9974 0.9992  
## Pos Pred Value 0.9964 0.9993 0.9920 0.9868 0.9965  
## Neg Pred Value 0.9998 0.9976 0.9983 0.9982 0.9995  
## Prevalence 0.2845 0.1935 0.1744 0.1639 0.1838  
## Detection Rate 0.2843 0.1916 0.1730 0.1624 0.1834  
## Detection Prevalence 0.2854 0.1917 0.1744 0.1645 0.1840  
## Balanced Accuracy 0.9991 0.9950 0.9951 0.9940 0.9986

plot(gbmFit1, ylim=c(0.9, 1))



#Predicting Results on the Test Data  
predictionB2 <- predict(modFitB1, testing, type = "class")  
predictionB2

## 1 2 3 41 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

comment: The Random Forrest gave the most accuracy with 99.89% accuracy than the Decision Trees and the Generalized Boosted Regression. The out of sample rate is .11%