Machine Learning Final Project

Edward S.

November 5, 2017

setwd("C:/Users/edwsp/Desktop/MACHINE LEARNING")  
set.seed(1986)  
library(knitr)  
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(rpart)  
library(rpart.plot)  
library(rattle)

## Rattle: A free graphical interface for data science with R.  
## Version 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

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:rattle':  
##   
## importance

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

# Obtain data:

if(!file.exists("training.csv")){  
 dir.create("training.csv")  
 train\_url <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"  
 download.file(url=train\_url, destfile="training.csv")  
 date.Downloaded.training<-date()}  
  
if(!file.exists("testing.csv")){  
 dir.create("testing.csv")  
 test\_url <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"  
 download.file(url=test\_url, destfile="testing.csv")  
 date.Downloaded.testing<-date()}  
  
train <- read.csv("training.csv", na.strings=c("NA","#DIV/0!",""))  
test <- read.csv("testing.csv", na.strings=c("NA","#DIV/0!",""))

# Examine the data: I started out by looking much more closely than included below, just listed the short output things.

#names(train)  
dim(train)

## [1] 19622 160

dim(test)

## [1] 20 160

summary(train$classe)

## A B C D E   
## 5580 3797 3422 3216 3607

# Partition the training dataset into a training and test set..

inTrain <- createDataPartition(train$classe, p=3/4, list=FALSE)  
myTrain <- train[inTrain,]  
myTest <- train[-inTrain,]

# Data clean-up: remove if >50%NA, no variability, or not meaningful (i.e. labels)- takes us down to ~52 usable variables

myTrain->Tr  
dim(Tr)

## [1] 14718 160

nzv <- nearZeroVar(Tr, saveMetrics=TRUE)  
Tr <- Tr[,nzv$nzv==FALSE]  
dim(Tr)

## [1] 14718 127

Tr<-Tr[,-which(colMeans(is.na(Tr))>0.5)]  
dim(Tr)

## [1] 14718 59

drop<-names(Tr) %in% c("X","raw\_timestamp\_part\_1","user\_name","raw\_timestamp\_part\_2","cvtd\_timestamp", "new\_window", "num\_window")  
Tr<-Tr[!drop]  
dim(Tr)

## [1] 14718 53

# Build a Simple Random Forest Model and determine in Sample Accuracy as ~99.9%:

rfMod<-randomForest(classe~., data=Tr)  
predictTrain <- predict(rfMod, myTrain, type = "class")  
confusionMatrix(myTrain$classe, predictTrain)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 4185 0 0 0 0  
## B 0 2848 0 0 0  
## C 0 0 2567 0 0  
## D 0 0 0 2412 0  
## E 0 0 0 0 2706  
##   
## Overall Statistics  
##   
## Accuracy : 1   
## 95% CI : (0.9997, 1)  
## No Information Rate : 0.2843   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 1   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 1.0000 1.0000 1.0000 1.0000 1.0000  
## Specificity 1.0000 1.0000 1.0000 1.0000 1.0000  
## Pos Pred Value 1.0000 1.0000 1.0000 1.0000 1.0000  
## Neg Pred Value 1.0000 1.0000 1.0000 1.0000 1.0000  
## Prevalence 0.2843 0.1935 0.1744 0.1639 0.1839  
## Detection Rate 0.2843 0.1935 0.1744 0.1639 0.1839  
## Detection Prevalence 0.2843 0.1935 0.1744 0.1639 0.1839  
## Balanced Accuracy 1.0000 1.0000 1.0000 1.0000 1.0000

# Apply the RF Model on the Partitioned Test Set and Determine OUt of Sample Accuracy: Here, Out of Sample Accuracy on Test set= ~99.9%, which is suprisingly damn good. As this is so good, no reason to go back and do more cross-validation, look at alternative models, etc. I'm satisfied.

predictTest <- predict(rfMod, myTest, type = "class")  
confusionMatrix(myTest$classe, predictTest)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1391 4 0 0 0  
## B 3 944 2 0 0  
## C 0 10 842 3 0  
## D 0 0 9 794 1  
## E 0 0 0 3 898  
##   
## Overall Statistics  
##   
## Accuracy : 0.9929   
## 95% CI : (0.9901, 0.995)  
## No Information Rate : 0.2843   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.991   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9978 0.9854 0.9871 0.9925 0.9989  
## Specificity 0.9989 0.9987 0.9968 0.9976 0.9993  
## Pos Pred Value 0.9971 0.9947 0.9848 0.9876 0.9967  
## Neg Pred Value 0.9991 0.9965 0.9973 0.9985 0.9998  
## Prevalence 0.2843 0.1954 0.1739 0.1631 0.1833  
## Detection Rate 0.2836 0.1925 0.1717 0.1619 0.1831  
## Detection Prevalence 0.2845 0.1935 0.1743 0.1639 0.1837  
## Balanced Accuracy 0.9984 0.9921 0.9919 0.9950 0.9991

# Now use the 20 Observation "Validation" Imported test set: Only based on 20 Observations but should be 100% correct.

predictTEST20 <- predict(rfMod, test, type = "class")  
predictTEST20

## 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