ML\_8\_Project

Karthik Vadhri

December 2, 2017

# Practical Machine Learning - Course Project

### Karthik Vadhri

# Practical Machine Learning Project : Prediction Assignment Writeup

## Overview

This document is the final report of the Peer Assessment project from Coursera’s course Practical Machine Learning, as part of the Specialization in Data Science. It was built up in RStudio, using its knitr functions, meant to be published in html format. This analysis meant to be the basis for the course quiz and a prediction assignment writeup. The main goal of the project is to predict the manner in which 6 participants performed some exercise as described below. This is the “classe” variable in the training set. The machine learning algorithm described here is applied to the 20 test cases available in the test data and the predictions are submitted in appropriate format to the Course Project Prediction Quiz for automated grading.

## 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 dumbell 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).

Read more: <http://groupware.les.inf.puc-rio.br/har#ixzz3xsbS5bVX>

## Data Loading and Exploratory Analysis

### Dataset Overview

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 <http://groupware.les.inf.puc-rio.br/har>. Full source:

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. “Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human ’13)”. Stuttgart, Germany: ACM SIGCHI, 2013.

My special thanks to the above mentioned authors for being so generous in allowing their data to be used for this kind of assignment.

A short description of the datasets content from the authors’ website:

“Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E).

Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. Participants were supervised by an experienced weight lifter to make sure the execution complied to the manner they were supposed to simulate. The exercises were performed by six male participants aged between 20-28 years, with little weight lifting experience. We made sure that all participants could easily simulate the mistakes in a safe and controlled manner by using a relatively light dumbbell (1.25kg)."

### Setting the environment

rm(list=ls()) # free up memory for the download of the data sets  
library(knitr)  
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(rpart)  
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 3.3.3

#library(rattle)  
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

library(corrplot)

## Warning: package 'corrplot' was built under R version 3.3.3

### Data Loading and Cleaning

UrlTrain <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"  
UrlTest <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"  
  
# download the datasets  
training <- read.csv(url(UrlTrain))  
testing <- read.csv(url(UrlTest))  
  
# create a partition with the training dataset   
inTrain <- createDataPartition(training$classe, p=0.7, list=FALSE)  
TrainSet <- training[inTrain, ]  
TestSet <- training[-inTrain, ]  
dim(TrainSet)

## [1] 13737 160

dim(TestSet)

## [1] 5885 160

#### Remove Variables with Zero Variance, and high NA's

NZV <- nearZeroVar(TrainSet)  
TrainSet <- TrainSet[, -NZV]  
TestSet <- TestSet[, -NZV]  
dim(TrainSet)

## [1] 13737 105

AllNA <- sapply(TrainSet, function(x) mean(is.na(x))) > 0.95  
TrainSet <- TrainSet[, AllNA==FALSE]  
TestSet <- TestSet[, AllNA==FALSE]  
dim(TrainSet)

## [1] 13737 59

dim(TestSet)

## [1] 5885 59

# remove variables that are Unique Identifiers (columns 1 to 5)  
TrainSet <- TrainSet[, -(1:5)]  
TestSet <- TestSet[, -(1:5)]  
dim(TrainSet)

## [1] 13737 54

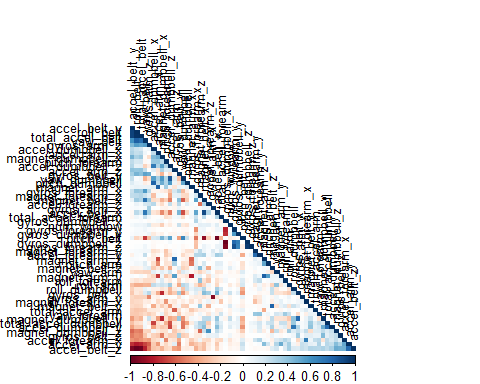
dim(TestSet)

## [1] 5885 54

### Correlation Analysis

We use the correlation plot to observe the relation among the variables.

corMatrix <- cor(TrainSet[, -54])  
corrplot(corMatrix, order = "FPC", method = "color", type = "lower",   
 tl.cex = 0.8, tl.col = rgb(0, 0, 0))



### Prediction Model Building

We shall use 3 different methods, and compare the accuracy of each one of them, before deciding the final model. Confusion matrix of each model is also plotted, to give a holistic picture of the model .

#### Method 1 : Random Forest

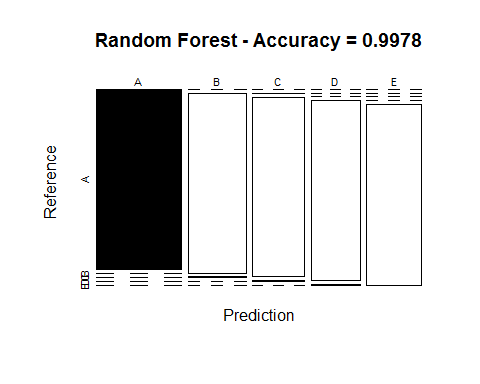
set.seed(12345)  
controlRF <- trainControl(method="cv", number=3, verboseIter=FALSE)  
modFitRandForest <- train(classe ~ ., data=TrainSet, method="rf",  
 trControl=controlRF)  
modFitRandForest$finalModel

##   
## Call:  
## randomForest(x = x, y = y, mtry = param$mtry)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 27  
##   
## OOB estimate of error rate: 0.25%  
## Confusion matrix:  
## A B C D E class.error  
## A 3905 0 0 0 1 0.0002560164  
## B 4 2650 3 1 0 0.0030097818  
## C 0 4 2391 1 0 0.0020868114  
## D 0 0 11 2240 1 0.0053285968  
## E 0 1 1 6 2517 0.0031683168

# prediction on Test dataset  
predictRandForest <- predict(modFitRandForest, newdata=TestSet)  
confMatRandForest <- confusionMatrix(predictRandForest, TestSet$classe)  
confMatRandForest

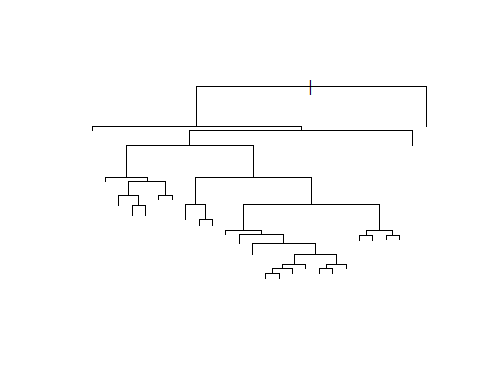
## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1674 0 0 0 0  
## B 0 1138 3 0 0  
## C 0 1 1023 5 0  
## D 0 0 0 959 4  
## E 0 0 0 0 1078  
##   
## Overall Statistics  
##   
## Accuracy : 0.9978   
## 95% CI : (0.9962, 0.9988)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9972   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 1.0000 0.9991 0.9971 0.9948 0.9963  
## Specificity 1.0000 0.9994 0.9988 0.9992 1.0000  
## Pos Pred Value 1.0000 0.9974 0.9942 0.9958 1.0000  
## Neg Pred Value 1.0000 0.9998 0.9994 0.9990 0.9992  
## Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Detection Rate 0.2845 0.1934 0.1738 0.1630 0.1832  
## Detection Prevalence 0.2845 0.1939 0.1749 0.1636 0.1832  
## Balanced Accuracy 1.0000 0.9992 0.9979 0.9970 0.9982

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



#### Method: Decision Trees

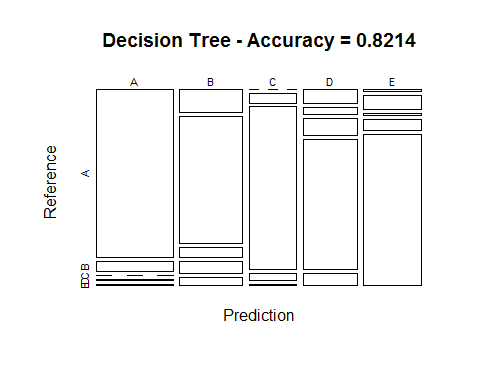
modFitDecTree <- rpart(classe ~ ., data=TrainSet, method="class")  
plot(modFitDecTree)



predictDecTree <- predict(modFitDecTree, newdata=TestSet, type="class")  
confMatDecTree <- confusionMatrix(predictDecTree, TestSet$classe)  
confMatDecTree

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1406 87 0 12 5  
## B 163 874 68 83 56  
## C 0 52 841 40 2  
## D 87 42 100 763 69  
## E 18 84 17 66 950  
##   
## Overall Statistics  
##   
## Accuracy : 0.8214   
## 95% CI : (0.8114, 0.8311)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7749   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.8399 0.7673 0.8197 0.7915 0.8780  
## Specificity 0.9753 0.9220 0.9807 0.9394 0.9615  
## Pos Pred Value 0.9311 0.7026 0.8995 0.7191 0.8370  
## Neg Pred Value 0.9387 0.9429 0.9626 0.9583 0.9722  
## Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Detection Rate 0.2389 0.1485 0.1429 0.1297 0.1614  
## Detection Prevalence 0.2566 0.2114 0.1589 0.1803 0.1929  
## Balanced Accuracy 0.9076 0.8447 0.9002 0.8655 0.9197

plot(confMatDecTree$table, col = confMatDecTree$byClass,   
 main = paste("Decision Tree - Accuracy =",  
 round(confMatDecTree$overall['Accuracy'], 4)))



### Method 3 : Generalized Boosted Model

controlGBM <- trainControl(method = "repeatedcv", number = 5, repeats = 1)  
modFitGBM <- train(classe ~ ., data=TrainSet, method = "gbm",  
 trControl = controlGBM, verbose = FALSE)

## Loading required package: gbm

## Loading required package: survival

## Warning: package 'survival' was built under R version 3.3.3

##   
## 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

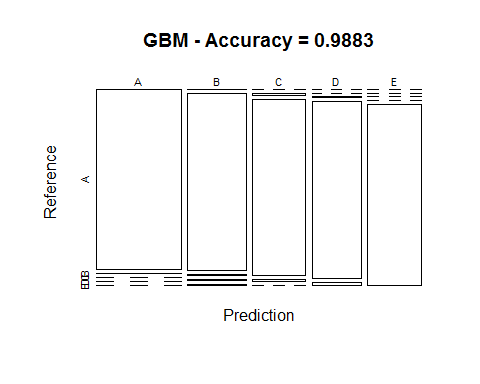
modFitGBM$finalModel

## A gradient boosted model with multinomial loss function.  
## 150 iterations were performed.  
## There were 53 predictors of which 43 had non-zero influence.

predictGBM <- predict(modFitGBM, newdata=TestSet)  
confMatGBM <- confusionMatrix(predictGBM, TestSet$classe)  
confMatGBM

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1670 3 0 0 0  
## B 4 1123 9 7 4  
## C 0 13 1014 10 0  
## D 0 0 3 947 16  
## E 0 0 0 0 1062  
##   
## Overall Statistics  
##   
## Accuracy : 0.9883   
## 95% CI : (0.9852, 0.9909)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9852   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9976 0.9860 0.9883 0.9824 0.9815  
## Specificity 0.9993 0.9949 0.9953 0.9961 1.0000  
## Pos Pred Value 0.9982 0.9791 0.9778 0.9803 1.0000  
## Neg Pred Value 0.9991 0.9966 0.9975 0.9965 0.9959  
## Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Detection Rate 0.2838 0.1908 0.1723 0.1609 0.1805  
## Detection Prevalence 0.2843 0.1949 0.1762 0.1641 0.1805  
## Balanced Accuracy 0.9984 0.9904 0.9918 0.9893 0.9908

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



## Applying the Selected Model to the Test Data

The accuracy of the 3 regression modeling methods above are:

Random Forest : 0.9963 Decision Tree : 0.7368 GBM : 0.9839

It is clearly evident that the RF seems to be better classifier, so we use that for the test sets given for the exercise.