Prediction Assignment

Francesco Mio

Prediction Assignment Course Project

Executive Summary

One thing that people of the quantified self movement regularly do is to quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, the 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. The goal of the project is to predict the manner in which they did the exercise.

Load libraries and data

```
library(caret)
```

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

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(randomForest)
```

```
## randomForest 4.6-14
## Type rfNews() to see new features/changes/bug fixes.
```

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

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

```
library(rattle)
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.3.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

##
## Attaching package: 'rattle'

## The following object is masked from 'package:randomForest':
##
##     importance
```

```
library(rpart)
library(rpart.plot)

train <- read.csv("pml-training.csv", na.strings=c("#DIV/0!", "NA", ""))
test <- read.csv("pml-testing.csv", na.strings=c("#DIV/0!", "NA", ""))

dim(train)
```

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

```
dim(test)
```

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

Data cleaning

Data cleaning, by removing all columns that contain

NAs or empty values. Also, I remove the first columns

that contain data, that won't help the predoction (see data

```
train_clean <- train[,colSums(is.na(train))==0]
train_clean <- train_clean[,-c(1:7)]

test_clean <- test[,colSums(is.na(test))==0]
test_clean <- test_clean[,-c(1:7)]

dim(train_clean)
```

summary in appendix 1 - i.e. timestamp data).

```
## [1] 19622    53
```

```
dim(test_clean)
```

```
## [1] 20 53
```

```
nearZeroVariables <- nearZeroVar(train_clean)
nearZeroVariables
```

```
## integer(0)
```

Create validation set

```
train_partition <- createDataPartition(train_clean$classe, p=0.8, list=FALSE)
train_final <- train_clean[train_partition,]
valid_final <- train_clean[-train_partition,]

test_final <- test_clean

dim(train_final)
```

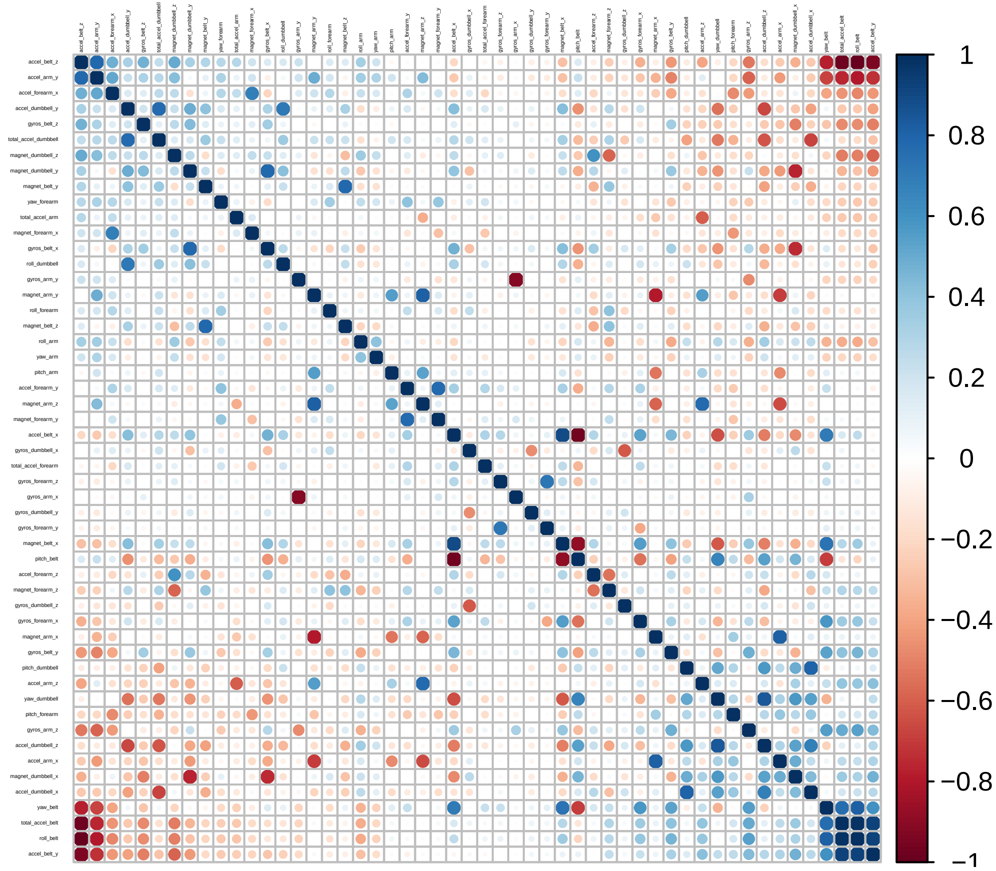
```
## [1] 15699    53
```

```
dim(valid_final)
```

```
## [1] 3923   53
```

Correlation Matrix

```
exerCorrmatrix<-cor(train_final[sapply(train_final, is.numeric)])
corrplot(exerCorrmatrix,order="FPC", tl.cex=0.2, tl.col="black")
```

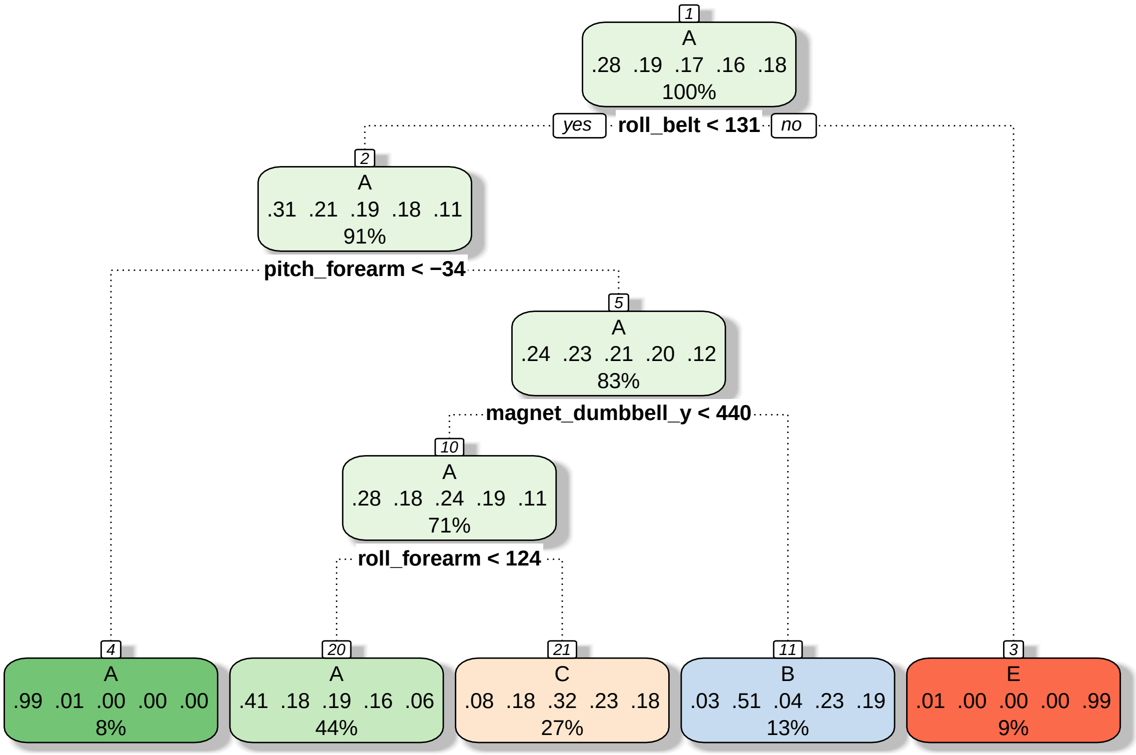


Cross Validation

```
cross_validation <- trainControl(method='cv', number = 3)
```

Decision Tree

```
set.seed(111)
decisionTree_model <- train(classe~., data=train_final, method="rpart", trControl=cross_validation)
fancyRpartPlot(decisionTree_model$finalModel)
```



Rattle 2020−Feb−19 12:33:32 fmio

##

Decision Tree Model Performance

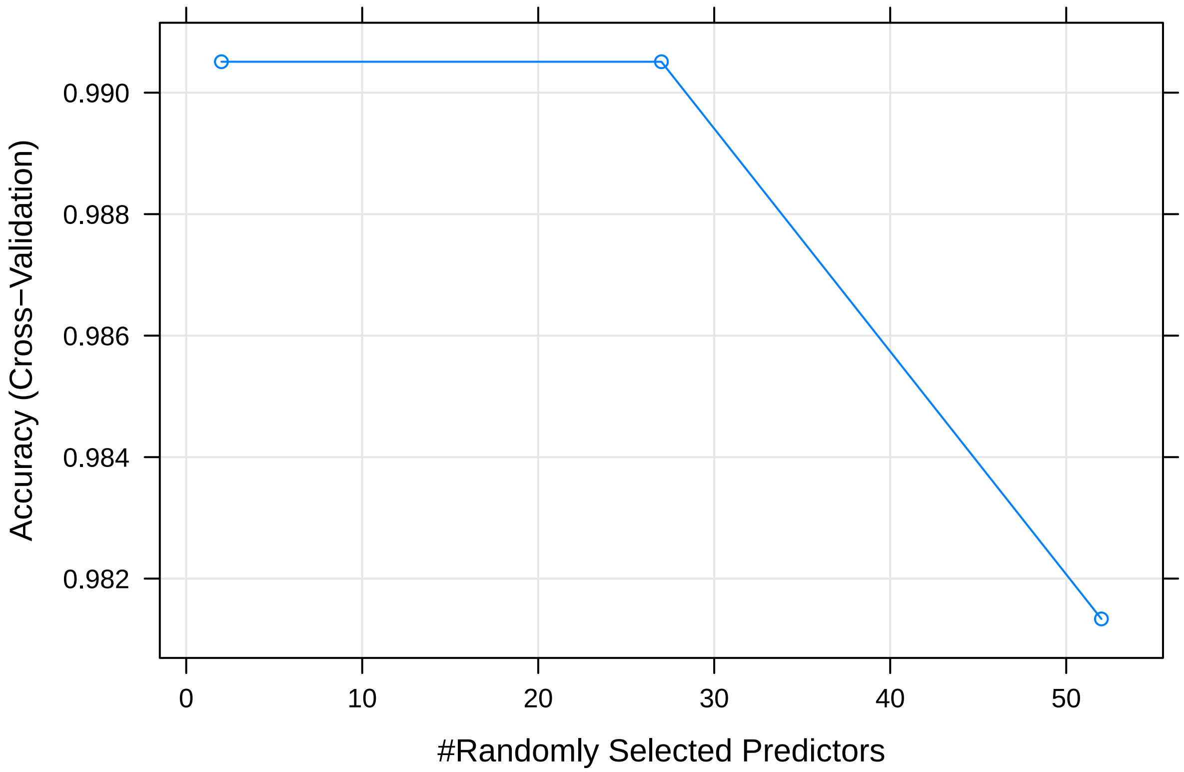
```
decisionTree_prediction <- predict(decisionTree_model,newdata=valid_final)
decisionTree_cm <- confusionMatrix(valid_final$classe,decisionTree_prediction)
decisionTree_cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1011   23   80    0    2
##          B  305  279  175    0    0
##          C  296   19  369    0    0
##          D  308  116  219    0    0
##          E  120  104  194    0  303
##
## Overall Statistics
##
##                Accuracy : 0.5001
##                  95% CI : (0.4844, 0.5159)
##     No Information Rate : 0.52
##     P-Value [Acc > NIR] : 0.9939
##
##                   Kappa : 0.3466
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
```

```
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.4956  0.51571  0.35583       NA  0.99344
## Specificity            0.9442  0.85807  0.89085   0.8361  0.88447
## Pos Pred Value         0.9059  0.36759  0.53947       NA  0.42025
## Neg Pred Value         0.6334  0.91719  0.79376       NA  0.99938
## Prevalence             0.5200  0.13790  0.26434   0.0000  0.07775
## Detection Rate         0.2577  0.07112  0.09406   0.0000  0.07724
## Detection Prevalence   0.2845  0.19347  0.17436   0.1639  0.18379
## Balanced Accuracy      0.7199  0.68689  0.62334       NA  0.93895
```

Random Forest

```r
set.seed(112)
randomForest_model <- train(classe~., data=train_final, method="rf", trControl=cross_validation, verbos
plot(randomForest_model)
```



#

Random Forest Model Performance

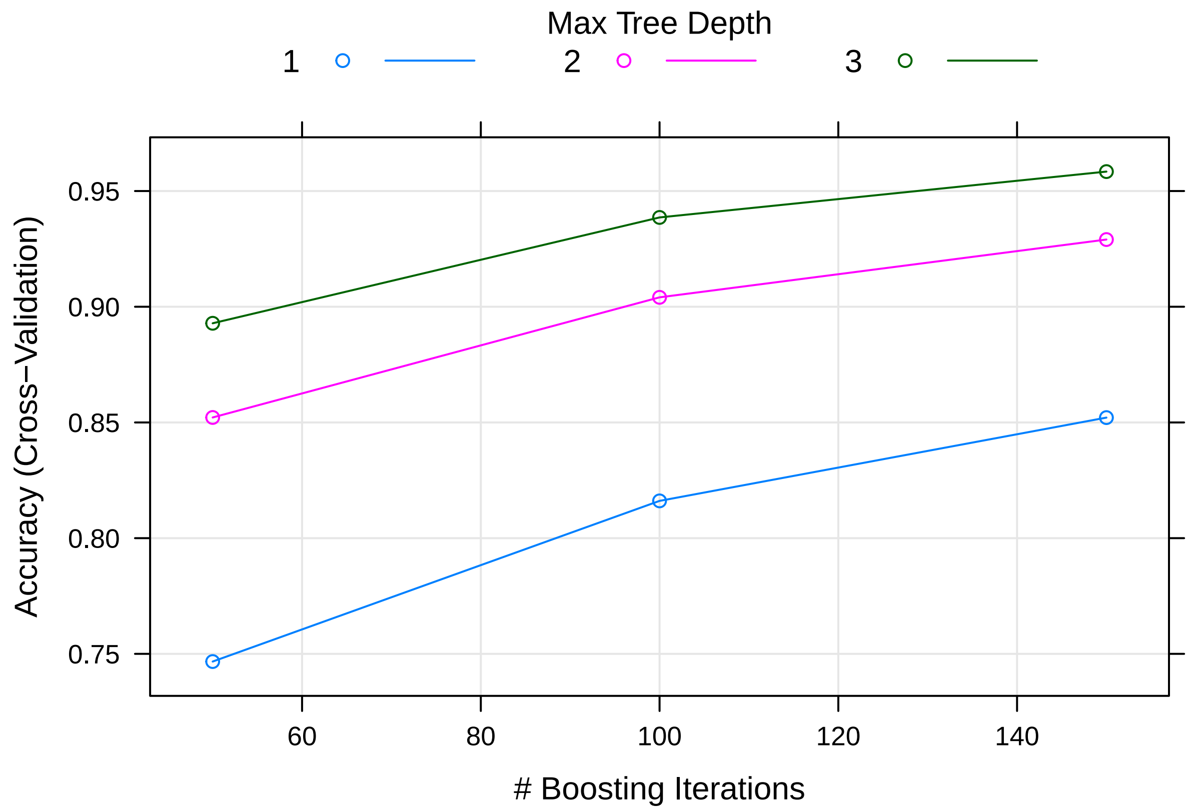
```r
randomForest_prediction <- predict(randomForest_model,newdata=valid_final)
randomForest_cm <- confusionMatrix(valid_final$classe,randomForest_prediction)
randomForest_cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1115    0    0    0    1
##          B    2  757    0    0    0
```

```
##          C    0    6  677    1    0
##          D    0    0    7  636    0
##          E    0    0    0    1  720
##
## Overall Statistics
##
##                Accuracy : 0.9954
##                  95% CI : (0.9928, 0.9973)
##     No Information Rate : 0.2847
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9942
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9982   0.9921   0.9898   0.9969   0.9986
## Specificity            0.9996   0.9994   0.9978   0.9979   0.9997
## Pos Pred Value         0.9991   0.9974   0.9898   0.9891   0.9986
## Neg Pred Value         0.9993   0.9981   0.9978   0.9994   0.9997
## Prevalence             0.2847   0.1945   0.1744   0.1626   0.1838
## Detection Rate         0.2842   0.1930   0.1726   0.1621   0.1835
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9989   0.9958   0.9938   0.9974   0.9992
```

Generalized Boosted Regression

```
set.seed(113)
gbm_model <- train(classe~., data=train_final, method="gbm", trControl=cross_validation, verbose=FALSE)
plot(gbm_model)
```



#

Generalized Boosted Regression Model Performance

```
gbm_prediction <- predict(gbm_model,newdata=valid_final)
gbm_cm <- confusionMatrix(valid_final$classe,gbm_prediction)
gbm_cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1104    5    6    1    0
##          B   22  717   19    1    0
##          C    0   26  655    2    1
##          D    0    2   20  621    0
##          E    1    8    7    8  697
##
## Overall Statistics
##
##                Accuracy : 0.9671
##                  95% CI : (0.961, 0.9725)
##     No Information Rate : 0.2873
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9584
##
##  Mcnemar's Test P-Value : 2.713e-08
##
## Statistics by Class:
##
```

```
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9796   0.9459   0.9264   0.9810   0.9986
## Specificity            0.9957   0.9867   0.9910   0.9933   0.9926
## Pos Pred Value         0.9892   0.9447   0.9576   0.9658   0.9667
## Neg Pred Value         0.9918   0.9870   0.9839   0.9963   0.9997
## Prevalence             0.2873   0.1932   0.1802   0.1614   0.1779
## Detection Rate         0.2814   0.1828   0.1670   0.1583   0.1777
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9876   0.9663   0.9587   0.9872   0.9956
```

Choosing the best Model

Comparing all three confusion matrices

to find the most accurate one.

The Random Forest has the highest "Accuracy"

value in the confusion matrix summary.

We will use the radomForest_model with

```
prediction_test <- predict(randomForest_model,newdata=test_final)
prediction_test
```

the test_final data set.

```
##  [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
```

Appendix

Appendix 1: Boxplot

```
summary(train)[,1:7]
```

```
##        X            user_name    raw_timestamp_part_1 raw_timestamp_part_2
##  Min.   :    1   adelmo  :3892   Min.   :1.322e+09    Min.   :   294
##  1st Qu.: 4906   carlitos:3112   1st Qu.:1.323e+09    1st Qu.:252912
##  Median : 9812   charles :3536   Median :1.323e+09    Median :496380
##  Mean   : 9812   eurico  :3070   Mean   :1.323e+09    Mean   :500656
##  3rd Qu.:14717   jeremy  :3402   3rd Qu.:1.323e+09    3rd Qu.:751891
##  Max.   :19622   pedro   :2610   Max.   :1.323e+09    Max.   :998801
```

```
##
##          cvtd_timestamp  new_window    num_window
##  28/11/2011 14:14: 1498   no :19216   Min.   :  1.0
##  05/12/2011 11:24: 1497   yes:  406   1st Qu.:222.0
##  30/11/2011 17:11: 1440               Median :424.0
##  05/12/2011 11:25: 1425               Mean   :430.6
##  02/12/2011 14:57: 1380               3rd Qu.:644.0
##  02/12/2011 13:34: 1375               Max.   :864.0
##  (Other)         :11007
```