Course Project

Hong Chen

5/28/2020

Background

In this project, our 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 training data for this project are available here: ## https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

##The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har).

```
library(knitr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(rpart)
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

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

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

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

```
## Warning: package 'rpart.plot' was built under R version 3.6.3
```

```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.6.3
```

```
## 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.
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.6.3
```

```
## randomForest 4.6-14
```

```
## 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
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 3.6.3
```

```
## corrplot 0.84 loaded
```

```
data1 = read.csv("~/Course 8/Practical-machine-learning-project/pml-training.csv")
data2 = read.csv("~/Course 8/Practical-machine-learning-project/pml-testing.csv")
##dim(data1)
##dim(data2)

## Diving original training set(19622 in total) to trainig (75%) and testing (25%)
inTrain = createDataPartition(data1$classe, p = 3/4)[[1]]
training = data1[ inTrain,]
testing = data1[-inTrain,]

## Both datasets have 160 variables. will remove Near Zero variance (NZV) variables, resulting in 102 pre
dictors after processing
NZV <- nearZeroVar(training)
training <- training[, -NZV]
testing  <- testing[, -NZV]
## dim(training)

## Furhter reducne # of predictors by removing variables that are mostly NA, resulting in 59 predictors l
eft
AllNA    <- sapply(training, function(x) mean(is.na(x))) > 0.95
training <- training[, AllNA==FALSE]
testing  <- testing[, AllNA==FALSE]
## dim(training)

## Further remove identification only variables (columns 1 to 7), resulting in 52 predictors only
training <- training[, -(1:7)]
testing  <- testing[, -(1:7)]
dim(training)
```

```
## [1] 14718    52
```

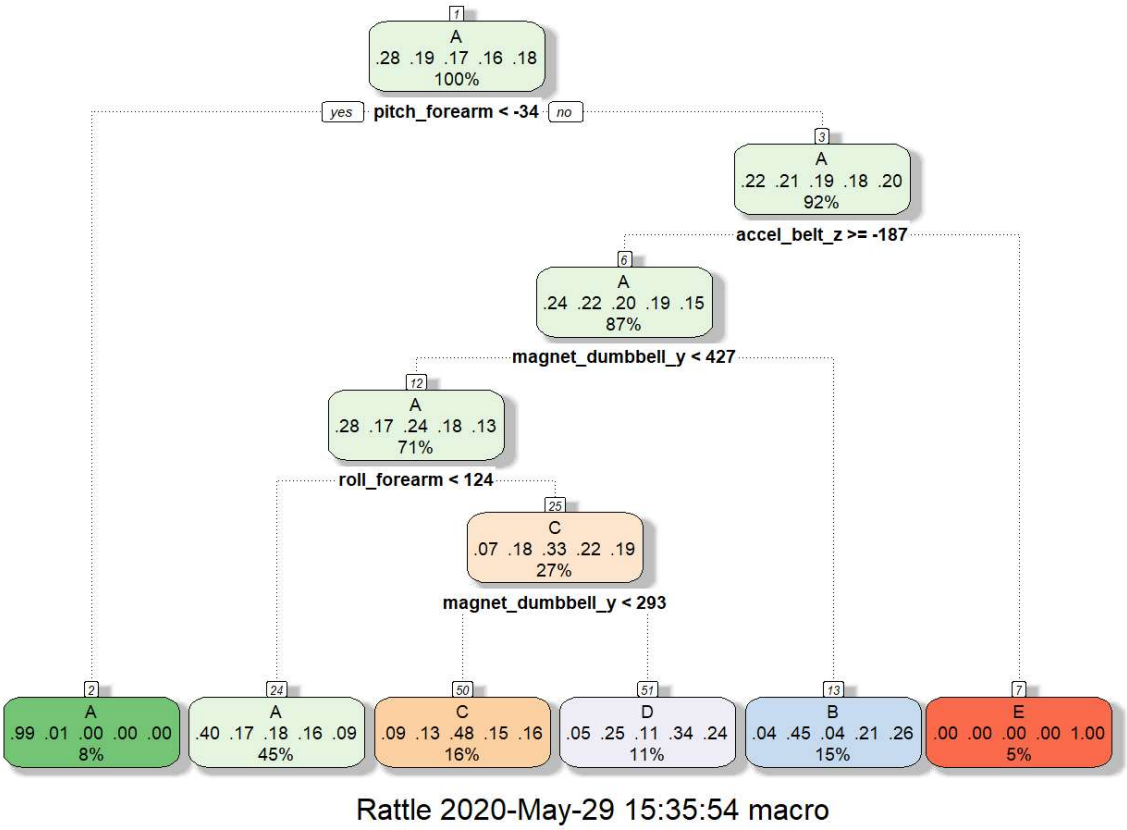
```
## Model Building will include fist Linear Discriminant Analysis (LDA), then non-linea indluding random f
orest (rf),  gradient boosting method (gbm), but I will implement cross-validation to avoid over-fitting
 first
set.seed(12345)
fitControl <- trainControl(method='cv', number = 3)

start1 <- Sys.time()
mod_rpart <- train(classe ~ ., data = training, method = "rpart")
end1 <- Sys.time()
time_rpart <- end1 - start1
mod_rpart
```

```
## CART
##
## 14718 samples
##    51 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 14718, 14718, 14718, 14718, 14718, 14718, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.03199468  0.4921366  0.3401518
##   0.03436818  0.4838101  0.3289876
##   0.06425045  0.3559468  0.1194586
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.03199468.
```

```
time_rpart
```

```
## Time difference of 18.26053 secs
```

```
fancyRpartPlot(mod_rpart$finalModel)
```



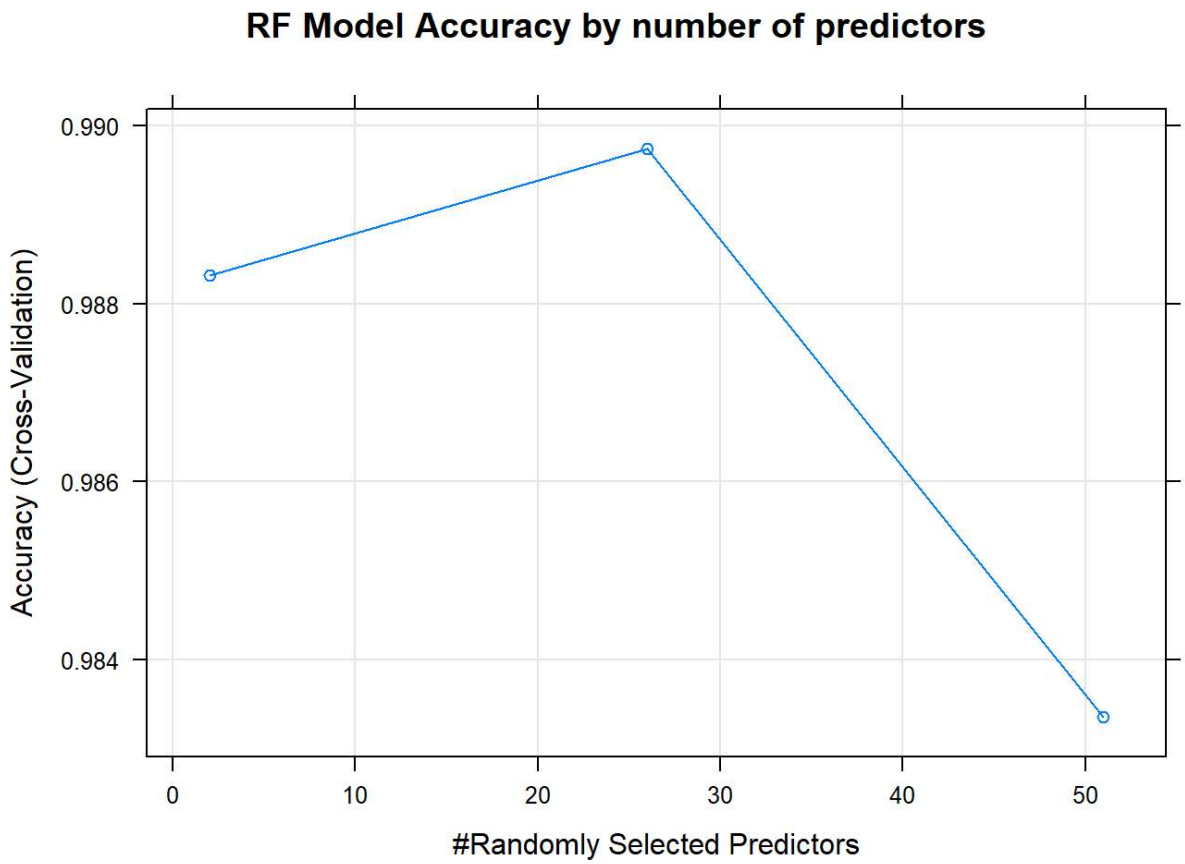
```
start1 <- Sys.time()
mod_rf <- train(classe ~ ., data = training, method = "rf",trControl=fitControl, VERBOSE=FALSE)
end1 <- Sys.time()
time_rf <- end1 - start1
mod_rf
```

```
## Random Forest
##
## 14718 samples
##    51 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (3 fold)
## Summary of sample sizes: 9812, 9812, 9812
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9883136  0.9852150
##   26    0.9897405  0.9870205
##   51    0.9833537  0.9789383
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 26.
```

```
time_rf
```

```
## Time difference of 7.28141 mins
```

```
plot(mod_rf,main="RF Model Accuracy by number of predictors")
```




```
start1 <- Sys.time()
mod_lda <- train(classe ~ ., data = training, method = "lda")
end1 <- Sys.time()
time_lda <- end1 - start1
time_lda
```

```
## Time difference of 10.65626 secs
```

```
mod_lda
```

```
## Linear Discriminant Analysis
##
## 14718 samples
##    51 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 14718, 14718, 14718, 14718, 14718, 14718, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6842871  0.6003504
```

```
pred_rf <- predict(mod_rf, testing)
pred_rpart <- predict(mod_rpart, testing)
pred_lda <- predict(mod_lda, testing)
#pred_gbm <- predict(mod_gbm, testing)

#predDF <- data.frame(pred_rf, pred_gbm, pred_lda, diagnosis = testing$classe)
#combModFit <- train(classe ~ ., method = "rf", data = predDF)
#combPred <- predict(combModFit, predDF)

confusionMatrix(pred_rf, testing$classe)$overall[1]
```

```
##  Accuracy
## 0.9932708
```

```
confusionMatrix(pred_rpart, testing$classe)$overall[1]
```

```
##  Accuracy
## 0.4859299
```

```
confusionMatrix(pred_lda, testing$classe)$overall[1]
```

```
##  Accuracy
## 0.6863785
```

```
#confusionMatrix(pred_gbm, testing$classe)$overall[1]
#confusionMatrix(combPred, testing$classe)$overall[1]

## Analysis
## It seems randon forrest has achieved close to 99% accuracy in both trainnig and test dataset. we will
 use this as final model for downloaded test dataset, otherwise we could try to use model combination to
 increase accuracy when needed
data2  <- data2[, -NZV]
data2  <- data2[, AllNA==FALSE]
data2  <- data2[, -(1:7)]
dim(data2)
```

```
## [1] 20 52
```

```
pred_final <- predict(mod_rf, data2)
pred_final
```

```
##  [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
```

Final anylysis

Although rf has the most accuracy predicton method, it is also the longest time to run wiht most resources required (7 min in this case compared with other 2 methods(lda and rpart around 15s). It will have bigger impact when data source is much bigger.