Practical Machine Learning - Prediction Assignment Writeup

Liew Chee Hau

1/17/2021

Overview

The goal of your project is to predict the manner in which they did the exercise. The machine learning algorithm, which uses the classe variable in the training set, is applied to the 20 test cases available in the test data.

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).

Data loading and Cleaning

Load the required R packages

```
library(knitr)
library(caret)

## Warning: package 'caret' was built under R version 4.0.3

## Loading required package: lattice

## Loading required package: ggplot2

library(rpart)
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 4.0.3

library(rattle)

## Warning: package 'rattle' was built under R version 4.0.3
## Loading required package: tibble

## Loading required package: bitops

## Rattle: A free graphical interface for data science with R.
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

library(randomForest)

## Warning: package 'randomForest' was built under R version 4.0.3

## 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
```

Read the datasets

```
urltrain <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
training.csv"
urltest  <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-
testing.csv"

training <- read.csv(url(urltrain))
testing  <- read.csv(url(urltest))
```

Create data 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
```

There are total 160 variables for both datasets. There are plenty of variables with NA. Near Zero Variance (NZV) removed these variables and the ID variables as well.

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

## [1] 13737   105

dim(TestSet)

## [1] 5885  105
```

Remove variables that are mostly NA

```
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 identification only variables (columns 1 to 5)

```
TrainSet <- TrainSet[, -(1:5)]
TestSet  <- TestSet[, -(1:5)]
dim(TrainSet)

## [1] 13737    54

dim(TestSet)

## [1] 5885   54
```

After the data cleaning process, the total number of variables used for analysis is 54 only.

Prediction Model Building

Three methods will be applied to the model the regressions of the Train dataset and the method with the highest accuracy will be applied to the Test dataset for the quiz predictions. The methods are: Random Forests, Decision Tree and Generalized Boosted Model.

A Confusion Matrix is plotted at the end of each analysis to better visualize the accuracy of the models.

1) Random Forest

Model fit

```
set.seed(1234)
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: 0.24%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3905    0    0    0    1 0.0002560164
## B    6 2648    3    1    0 0.0037622272
## C    0    7 2388    1    0 0.0033388982
## D    0    0   11 2241    0 0.0048845471
## E    0    0    0    3 2522 0.0011881188
```

Prediction on Test dataset

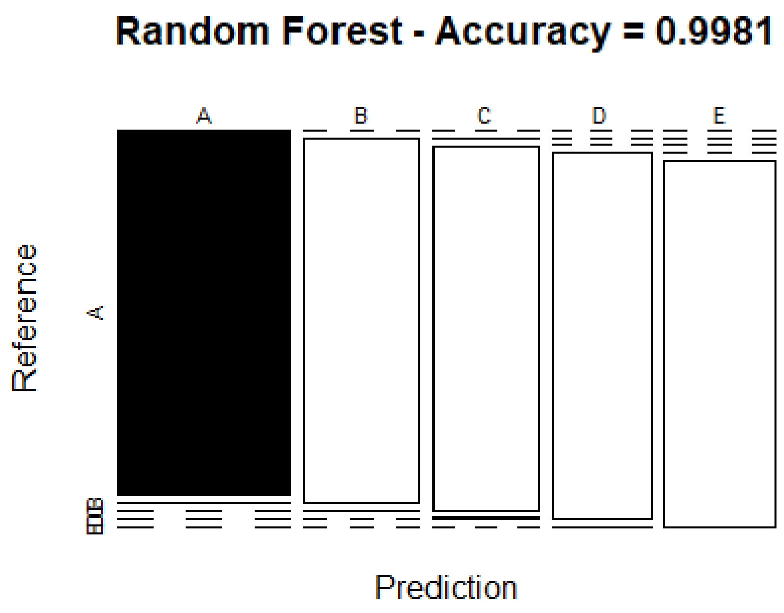
```
predictRandForest <- predict(modFitRandForest, newdata=TestSet)
confMatRandForest<-
confusionMatrix(predictRandForest,as.factor(TestSet$classe))
confMatRandForest
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    4    0    0    0
##          B    0 1133    2    0    0
##          C    0    2 1024    2    0
##          D    0    0    0  962    1
##          E    0    0    0    0 1081
##
## Overall Statistics
##
##                Accuracy : 0.9981
##                  95% CI : (0.9967, 0.9991)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9976
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9981   0.9979   0.9991
## Specificity            0.9991   0.9996   0.9992   0.9998   1.0000
## Pos Pred Value         0.9976   0.9982   0.9961   0.9990   1.0000
## Neg Pred Value         1.0000   0.9987   0.9996   0.9996   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1925   0.1740   0.1635   0.1837
## Detection Prevalence   0.2851   0.1929   0.1747   0.1636   0.1837
## Balanced Accuracy      0.9995   0.9972   0.9986   0.9989   0.9995
```

Plot results

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



2) Decision Trees

Model Fit

```
set.seed(1234)
modFitDecTree <- rpart(classe ~ ., data=TrainSet, method="class")
```

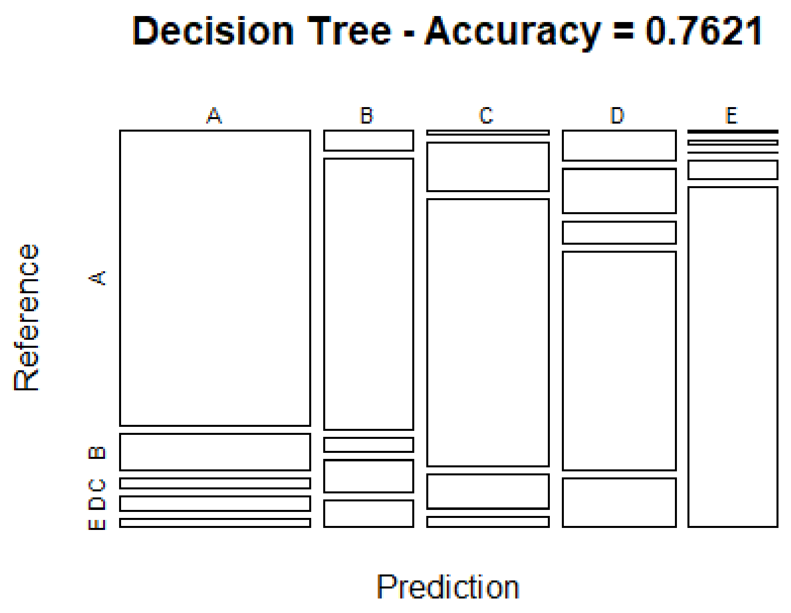
Prediction on Test dataset

```r
predictDecTree <- predict(modFitDecTree, newdata=TestSet, type="class")
confMatDecTree <- confusionMatrix(predictDecTree, as.factor(TestSet$classe))
confMatDecTree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1505  189   53   66   43
##          B   52  650   35   77   61
##          C   17  155  869  114   31
##          D   93  137   68  663  149
##          E    7    8    1   44  798
##
## Overall Statistics
##
##                Accuracy : 0.7621
##                  95% CI : (0.751, 0.7729)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6984
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8990   0.5707   0.8470   0.6878   0.7375
## Specificity            0.9166   0.9526   0.9348   0.9092   0.9875
## Pos Pred Value         0.8109   0.7429   0.7327   0.5973   0.9301
## Neg Pred Value         0.9581   0.9024   0.9666   0.9370   0.9435
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2557   0.1105   0.1477   0.1127   0.1356
## Detection Prevalence   0.3154   0.1487   0.2015   0.1886   0.1458
## Balanced Accuracy      0.9078   0.7616   0.8909   0.7985   0.8625
```

Plot results

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



3) Generalized Boosted Model

Model Fit

```
set.seed(1234)
controlGBM <- trainControl(method = "repeatedcv", number = 5, repeats = 1)
modFitGBM  <- train(classe ~ ., data=TrainSet, method = "gbm", trControl =
controlGBM, verbose = FALSE)
modFitGBM$finalModel

## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 53 predictors of which 52 had non-zero influence.
```

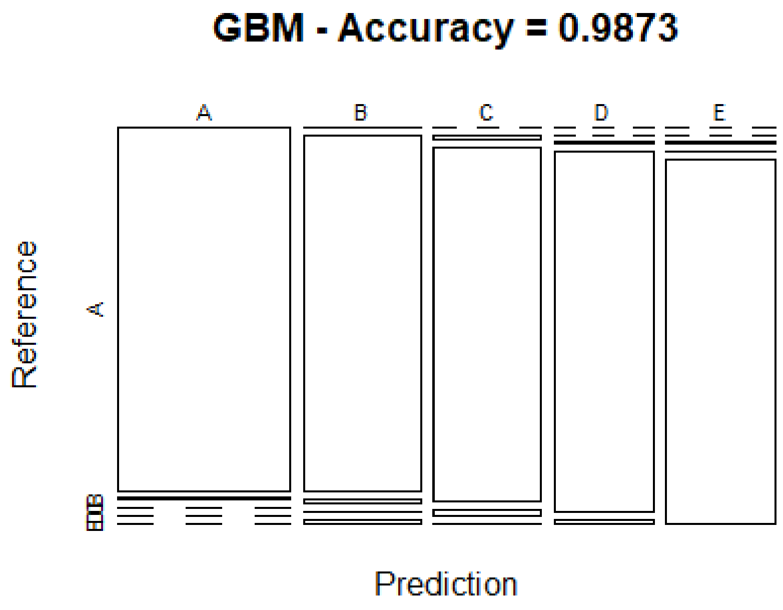
Prediction on Test dataset

```
predictGBM <- predict(modFitGBM, newdata=TestSet)
confMatGBM <- confusionMatrix(predictGBM, as.factor(TestSet$classe))
confMatGBM

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    5    0    0    0
##          B    1 1118   10    4   10
##          C    0   16 1013   14    1
##          D    0    0    2  945   10
##          E    0    0    1    1 1061
##
## Overall Statistics
##
##                Accuracy : 0.9873
##                  95% CI : (0.9841, 0.99)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9839
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9816   0.9873   0.9803   0.9806
## Specificity            0.9988   0.9947   0.9936   0.9976   0.9996
## Pos Pred Value         0.9970   0.9781   0.9703   0.9875   0.9981
## Neg Pred Value         0.9998   0.9956   0.9973   0.9961   0.9956
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1900   0.1721   0.1606   0.1803
## Detection Prevalence   0.2851   0.1942   0.1774   0.1626   0.1806
## Balanced Accuracy      0.9991   0.9881   0.9905   0.9889   0.9901
```

Plot results

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



Conclusion

The accuracy of the 3 regression modeling methods above are: 1.Random Forest:0.9981, 2.Decision Tree:0.7621, 3.GBM : 0.9873

From the above results, the Random Forest model has the best accuracy and hence will be used to predict the 20 quiz results.

```
predictTEST <- predict(modFitRandForest, newdata=testing)
predictTEST

##  [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
```