Practical Machine Learning Assignment

DTSK

1/19/2022

Introduction

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.The goal of this project is to predict the manner in which they did the exercise.

Preparation

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.1.2
```

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

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

```
library(ggplot2)
library(lattice)
library (rpart)
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.1.2
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.1.2
```

```
## 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(RColorBrewer)
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.1.2
```

```
## corrplot 0.92 loaded
```

Downloading of data

```
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
```

Data Preparation

Elimate Variables which are having nearly zero variance.

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

```
## [1] 13737   104
```

```
dim(TestSet)
```

```
## [1] 5885  104
```

Remove variable that are 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
```

Exclude col1 to col5 as they are not related to the model

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

```
## [1] 13737    54
```

```
dim(TestSet)
```

```
## [1] 5885   54
```

Random Forest

```
set.seed(111)
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 = min(param$mtry, ncol(x)))
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.17%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3905    1    0    0    0 0.0002560164
## B    8 2649    1    0    0 0.0033860045
## C    0    4 2391    1    0 0.0020868114
## D    0    0    6 2246    0 0.0026642984
## E    0    0    0    3 2522 0.0011881188
```

Prediction on test data

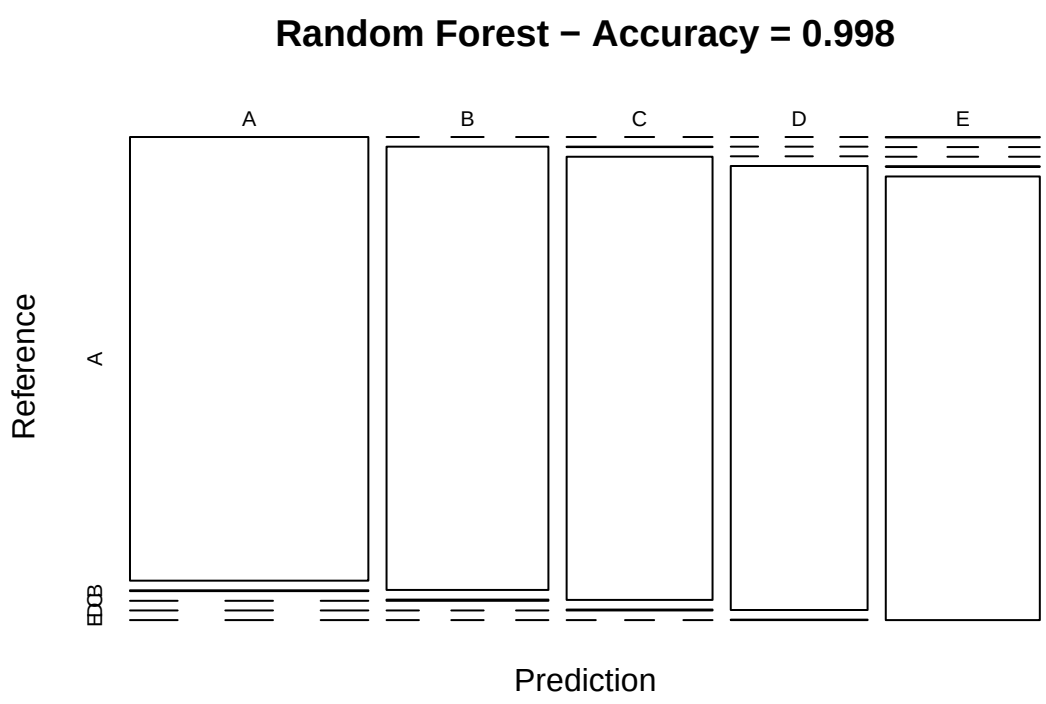
```
predictRandForest <- predict(modFitRandForest, newdata=TestSet)
confMatRandForest <- confusionMatrix(predictRandForest, as.factor(TestSet$classe))
confMatRandForest
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
```

```
##          A 1673    3    0    0    0
##          B    0 1135    3    0    0
##          C    0    1 1023    2    0
##          D    0    0    0  961    1
##          E    1    0    0    1 1081
##
## Overall Statistics
##
##                Accuracy : 0.998
##                  95% CI : (0.9964, 0.9989)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9974
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9965   0.9971   0.9969   0.9991
## Specificity            0.9993   0.9994   0.9994   0.9998   0.9996
## Pos Pred Value         0.9982   0.9974   0.9971   0.9990   0.9982
## Neg Pred Value         0.9998   0.9992   0.9994   0.9994   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1929   0.1738   0.1633   0.1837
## Detection Prevalence   0.2848   0.1934   0.1743   0.1635   0.1840
## Balanced Accuracy      0.9993   0.9979   0.9982   0.9983   0.9993
```

plot matrix results

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



Generalized Boosted Model

```
set.seed(111)
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 4.1.2
```

```
## Loaded gbm 2.1.8
```

```
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 53 had non-zero influence.
```

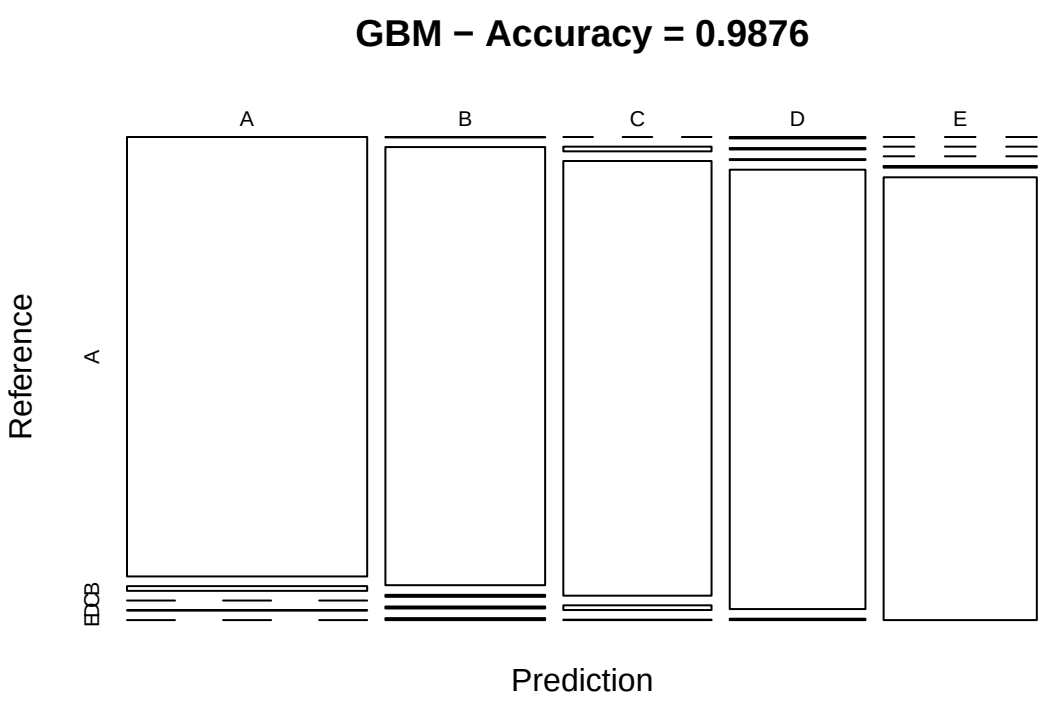
Prediction on test dataset

```
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 1670   18    0    1    0
##          B    1 1107    5    5    5
##          C    0   11 1019   11    1
##          D    3    3    2  943    3
##          E    0    0    0    4 1073
##
## Overall Statistics
##
##                Accuracy : 0.9876
##                  95% CI : (0.9844, 0.9903)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9843
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9976   0.9719   0.9932   0.9782   0.9917
## Specificity            0.9955   0.9966   0.9953   0.9978   0.9992
## Pos Pred Value         0.9888   0.9858   0.9779   0.9885   0.9963
## Neg Pred Value         0.9990   0.9933   0.9986   0.9957   0.9981
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2838   0.1881   0.1732   0.1602   0.1823
## Detection Prevalence   0.2870   0.1908   0.1771   0.1621   0.1830
## Balanced Accuracy      0.9965   0.9843   0.9942   0.9880   0.9954
```

plot matrix results

```
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 2 regression modeling methods above are: Random Forest : 0.9978 GBM : 0.9884 In that case, the Random Forest model will be applied to predict the quiz.

```
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
```