Practical_ML_Course_Project

WJC

25/10/2020

Instruction

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.

The goal of this project is to predict the manner in which they did the exercise. This is the `classe` variable in the training set.

Data description

The outcome variable is `classe`, a factor variable with 5 levels. Participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different ways:

Class A- exactly according to the specification Class B- throwing the elbows to the front Class C- lifting the dumbbell only halfway Class D- lowering the dumbbell only halfway Class E- throwing the hips to the front

Data Preprocessing

```
library(caret)
```

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

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

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

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

Download the Data

```
trainUrl <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
trainFile <- "./data/pml-training.csv"
testFile  <- "./data/pml-testing.csv"
if (!file.exists("./data")) {
  dir.create("./data")
}
if (!file.exists(trainFile)) {
  download.file(trainUrl, destfile=trainFile, method="curl")
}
if (!file.exists(testFile)) {
  download.file(testUrl, destfile=testFile, method="curl")
}
```

Read the Data

```
train.raw <- read.csv("./data/pml-training.csv")
test.raw <- read.csv("./data/pml-testing.csv")
dim(train.raw)
```

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

```
dim(test.raw)
```

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

There are 19622 observations and 160 variables in the training data set; 20 observations and 160 variables in the testing data set. The "classe" variable in the training set is the outcome to predict.

Clean the data

Clean the data and remove observations with missing values and irrelevant variables.

```
sum(complete.cases(train.raw))
```

```
## [1] 406
```

```
test.raw <- test.raw[, colSums(is.na(test.raw)) == 0]
train.raw <- train.raw[, colSums(is.na(train.raw)) == 0]


classe <- train.raw$classe
train.remove <- grepl("^X|timestamp|window", names(train.raw))
train.raw <- train.raw[, !train.remove]
train.cleaned <- train.raw[, sapply(train.raw, is.numeric)]
train.cleaned$classe <- classe
test.remove <- grepl("^X|timestamp|window", names(test.raw))
test.raw <- test.raw[, !test.remove]
test.cleaned <- test.raw[, sapply(test.raw, is.numeric)]
```

The cleaned training data set now has 19622 observations and 53 variables and the testing data set has 20 observations and 53 variables.

Slice the data

Split the cleaned training data set into the training data set (70%) and the validation data set (30%).

```
set.seed(251020)
Train <- createDataPartition(train.cleaned$classe, p=0.70, list=F)
train.data <- train.cleaned[Train, ]
test.data <- train.cleaned[-Train, ]
```

Data Modeling

Fit a prediction model for activity recognition using Random Forest algorithm (5-fold cross-validation) when applying the algorithm

```
controlrf <- trainControl(method="cv", 5)
modelrf <- train(classe ~ ., data=train.data, method="rf", trControl=controlrf, ntree=300)
modelrf
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10990, 10990, 10989, 10990, 10989
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9898087  0.9871073
##   27    0.9916287  0.9894101
##   52    0.9818736  0.9770668
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

Estimate the model performance on the validation data set.

```
predictrf <- predict(modelrf, test.data)
confusionMatrix(as.factor(test.data$classe), predictrf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1670    3    1    0    0
##          B    9 1126    4    0    0
##          C    0    6 1012    8    0
##          D    0    2   11  950    1
##          E    0    1    1    0 1080
##
## Overall Statistics
##
##                Accuracy : 0.992
##                  95% CI : (0.9894, 0.9941)
##     No Information Rate : 0.2853
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9899
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9946   0.9895   0.9835   0.9916   0.9991
## Specificity            0.9990   0.9973   0.9971   0.9972   0.9996
## Pos Pred Value         0.9976   0.9886   0.9864   0.9855   0.9982
## Neg Pred Value         0.9979   0.9975   0.9965   0.9984   0.9998
## Prevalence             0.2853   0.1934   0.1749   0.1628   0.1837
## Detection Rate         0.2838   0.1913   0.1720   0.1614   0.1835
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9968   0.9934   0.9903   0.9944   0.9993
```

```
accuracy <- postResample(predictrf, as.factor(test.data$classe))
accuracy
```

```
##  Accuracy     Kappa
## 0.9920136 0.9898970
```

```
se <- 1 - as.numeric(confusionMatrix(as.factor(test.data$classe), predictrf)$overall[1])
se
```

```
## [1] 0.007986406
```

The estimated model accuracy is 99.2% and the estimated out-of-sample error is 0.8%.

Predicting for Test Data Set

Apply the prediction model to the testing data set (the `problem_id` column was removed) and show the final result.

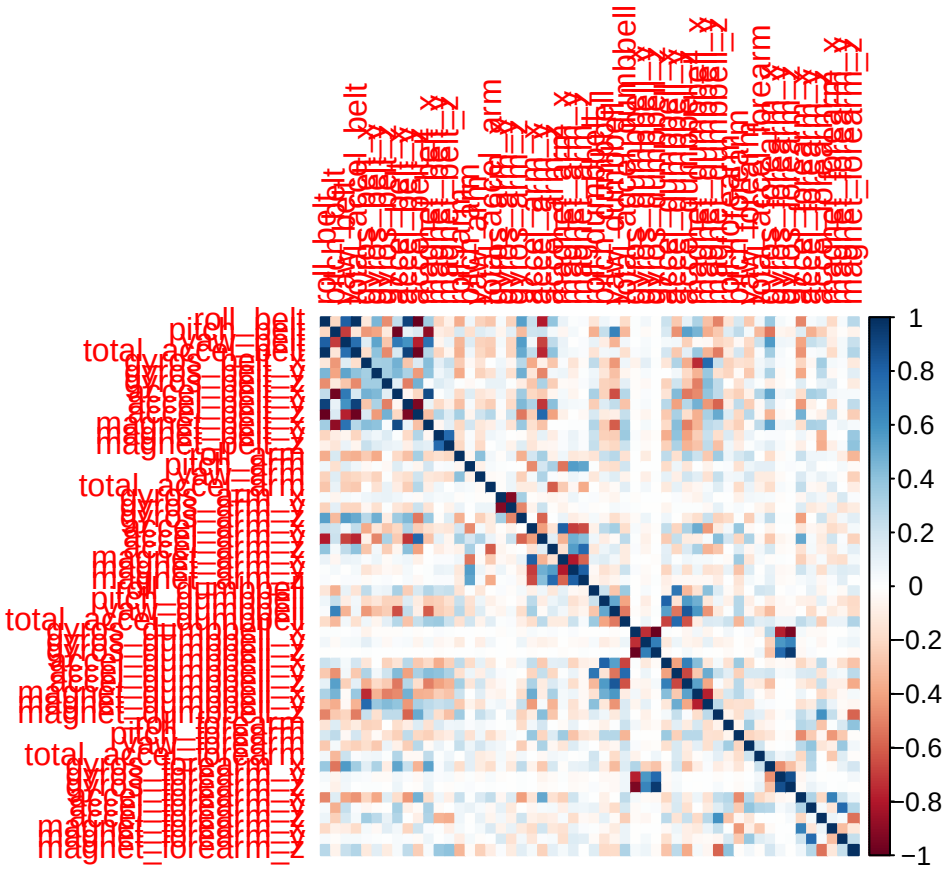
```
performance <- predict(modelrf, test.cleaned[, -length(names(test.cleaned))])
performance
```

```
##  [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: Figures

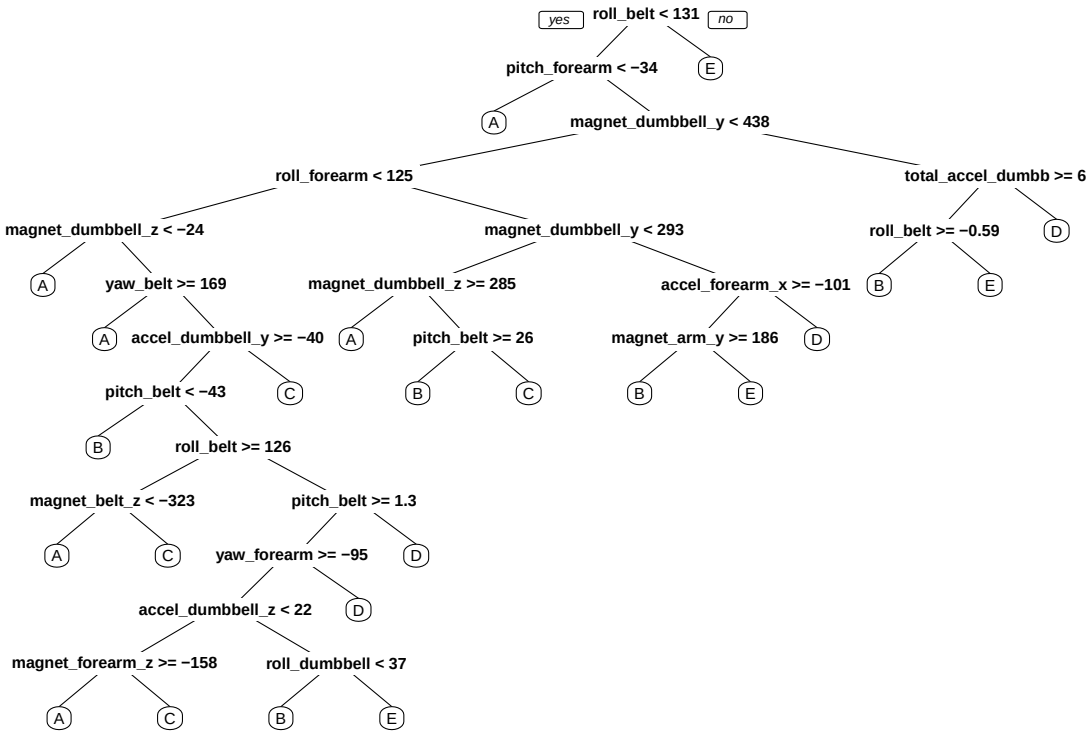
1. Correlation Matrix Visualisation

```
corr.plot <- cor(train.data[, -length(names(train.data))])
corrplot(corr.plot, method="color")
```



2. Decision Tree Visualisation

```
tree.model <- rpart(classe ~ ., data=train.data, method="class")
prp(tree.model)
```


yes
roll_belt < 131
no
pitch_forearm < −34
E
A
magnet_dumbbell_y < 438
roll_forearm < 125
total_accel_dumbb >= 6
magnet_dumbbell_z < −24
magnet_dumbbell_y < 293
roll_belt >= −0.59
D
A
yaw_belt >= 169
magnet_dumbbell_z >= 285
accel_forearm_x >= −101
B
E
A
accel_dumbbell_y >= −40
A
pitch_belt >= 26
magnet_arm_y >= 186
D
pitch_belt < −43
C
B
C
B
E
B
roll_belt >= 126
magnet_belt_z < −323
pitch_belt >= 1.3
A
C
yaw_forearm >= −95
D
accel_dumbbell_z < 22
D
magnet_forearm_z >= −158
roll_dumbbell < 37
A
C
B
E