practical machine learning

Project 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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Write up

The data provided consists of training and testing data. After importing the training and testing data, data cleaning was done. First 5 columns were removed as they do not predict the classe. Variables with NA and " " with a threshold of 0.95 were also removed. Besides that, near zero variance variables were also removed. This have reduced the number of variables from 160 to 54, including the predictor "classe".

The training data set was further splitted into training data and validation data, with a proportion of 70:30.

Then, random forest was used to build the model using all 53 variables to predict classe. There are a total of 5 classe: A, B, C, D and E. The model successfully predicted the classe of the training data and validation data with an accuracy of 1.

Lastly, the model was used to apply on the testing set to redict the classe. The result was shown below.

Load library

```
library(ggplot2)
library(caret)
library(AppliedPredictiveModeling)
library(randomForest)
library(dplyr)
library(rpart)
library(rpart.plot)
library(corrplot)
library(e1071)
```

Load data

```
## Load Data
## Create data directory
## Download and unzip data file
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")
}

## Load data
training <- read.csv("./data/pml-training.csv",na.strings=c("NA","#DIV/0!",""))
testing <- read.csv("./data/pml-testing.csv",na.strings=c("NA","#DIV/0!",""))
## Check size
dim(training)
```

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

```
dim(testing)
```

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

```
# Set seed
set.seed(1234)
```

Data Cleaning

Columns that contains NA and unnecessary variables should be removed

```
### Cleaning for testing set
# Removing first 5 columns which is not helpful
training <- training[, 6:dim(training)[2]]
# Setting threshold
total_rows <- dim(training)[1]
threshold <- 0.95 * total_rows
# Removing na columns based on the threshold of 0.95 and ""
useful_col <- !apply(training, 2, function (x) sum(is.na(x)) > threshold || sum(x=="") > threshold )
training <- training[,useful_col]
# Exclude near zero variance
nvz_col <- nearZeroVar(training, saveMetrics=TRUE)
training <- training[, nvz_col$nzv==FALSE]
training$classe <- as.factor(training$classe)
```

```
### Cleaning testing set
# Removing first 5 columns which is not helpful
testing <- testing[, 6:dim(testing)[2]]
# Removing na columns based on the threshold of 0.95 and ""
testing <- testing[,useful_col]
# Exclude near zero variance
testing <- testing[, nvz_col$nzv==FALSE]
dim(training);dim(testing)
```

```
## [1] 19622    54

## [1] 20 54
```

Data Partitioning for validation

create data partition for validation from training set

```
#Data partitioning for validation
inTrain <- createDataPartition(y=training$classe, p=0.7, list=FALSE)
training <- training[inTrain,]
validating <- training[-inTrain,]
```

Model developing

Training the model

```
# Training
mod1 <- randomForest(classe~., data=training)
# Predict
pred1 <- predict(mod1, training)
# Confusion Matrix
confusionMatrix(pred1, training$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3906    0    0    0    0
##          B    0 2658    0    0    0
##          C    0    0 2396    0    0
##          D    0    0    0 2252    0
##          E    0    0    0    0 2525
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9997, 1)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

Based on the confusion matrix, the model as an accuracy of 1.00 on the training set.

Validation test

```
pred2 <- predict(mod1, validating)
confusionMatrix(pred2,validating$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1165    0    0    0    0
##          B    0  787    0    0    0
##          C    0    0  738    0    0
##          D    0    0    0  672    0
##          E    0    0    0    0  761
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9991, 1)
##     No Information Rate : 0.2826
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000    1.000    1.000   1.0000
## Specificity            1.0000   1.0000    1.000    1.000   1.0000
## Pos Pred Value         1.0000   1.0000    1.000    1.000   1.0000
## Neg Pred Value         1.0000   1.0000    1.000    1.000   1.0000
## Prevalence             0.2826   0.1909    0.179    0.163   0.1846
## Detection Rate         0.2826   0.1909    0.179    0.163   0.1846
## Detection Prevalence   0.2826   0.1909    0.179    0.163   0.1846
## Balanced Accuracy      1.0000   1.0000    1.000    1.000   1.0000
```

Based on the confusion matrix, the accuracy for the model on validation is suprisingly 1.00

Testing set

```
testresults <- predict(mod1, newdata=testing)
testresults
```

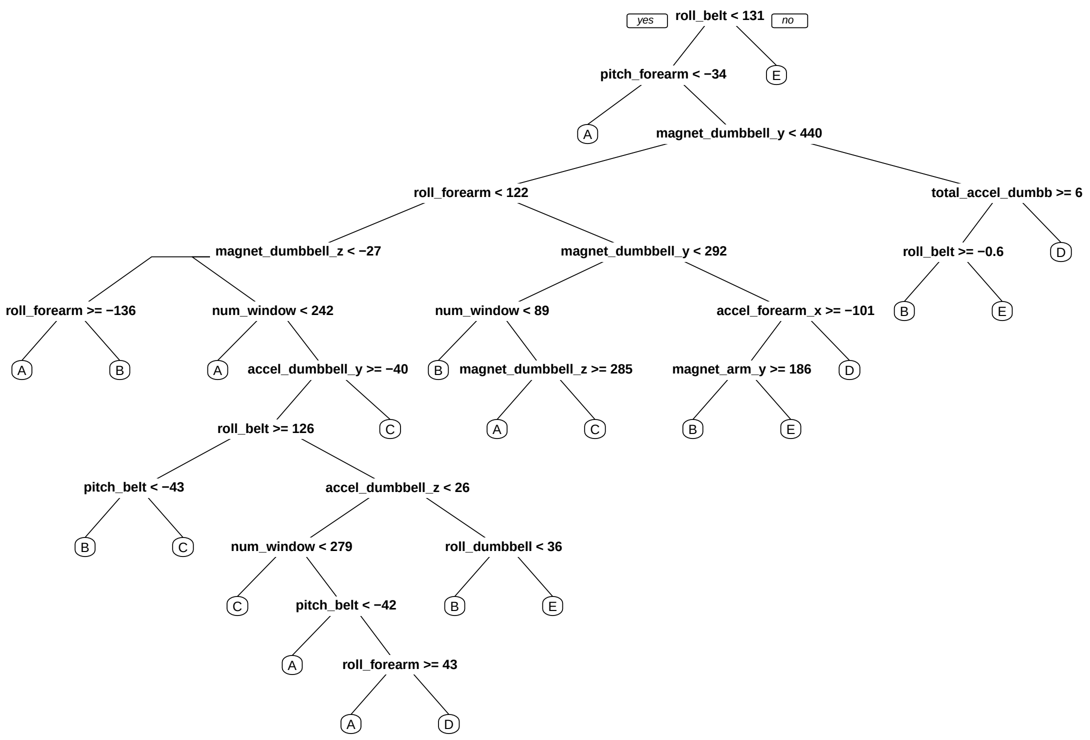
```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  B  A  A  E  D  B  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

The result above shows the predicted result for the testing set.

Appendix

Decision tree

```
treeModel <- rpart(classe ~ ., data=training, method="class")
prp(treeModel)
```


yes
roll_belt < 131
no
pitch_forearm < −34
E
A
magnet_dumbbell_y < 440
roll_forearm < 122
total_accel_dumbb >= 6
magnet_dumbbell_z < −27
magnet_dumbbell_y < 292
roll_belt >= −0.6
D
roll_forearm >= −136
num_window < 242
num_window < 89
accel_forearm_x >= −101
B
E
A
B
A
accel_dumbbell_y >= −40
B
magnet_dumbbell_z >= 285
magnet_arm_y >= 186
D
roll_belt >= 126
C
A
C
B
E
pitch_belt < −43
accel_dumbbell_z < 26
B
C
num_window < 279
roll_dumbbell < 36
C
pitch_belt < −42
B
E
A
roll_forearm >= 43
A
D