Practical Machine Learning Assignment

Chun-Fu Wang

Loading the Data

```
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(caret)
```

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

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

```
trainData <- read.csv("pml-training.csv")
testData <- read.csv("pml-testing.csv")

trainData <- trainData[2:length(trainData)]
testData <- testData[2:length(testData)]

dim(trainData)
```

```
## [1] 19622   159
```

```
dim(testData)
```

```
## [1]  20 159
```

There are `{r} length(trainData)` observations in the training data set and `{r} length(testData)` observations in the test data set that we are going to predict.

Exploratory Data Analysis

Removing the near zero variance features as well as the statistically insignificant features.

```
# cleaning up data
dim(trainData)
```

```
## [1] 19622   159
```

```
nzv <- nearZeroVar(trainData)
filteredTrainData <- trainData[, -nzv]
filteredTestData <- testData[, -nzv]

# removed statistically insignificant variables
filteredTrainData <-
    filteredTrainData %>%
    select(-c(user_name,
              raw_timestamp_part_1,
              raw_timestamp_part_2,
              cvtd_timestamp,
              max_roll_belt:var_yaw_belt,
              var_accel_arm,
              max_picth_arm:amplitude_yaw_arm,
              max_roll_dumbbell:amplitude_pitch_dumbbell,
              var_accel_dumbbell:var_yaw_dumbbell,
              max_picth_forearm:amplitude_pitch_forearm,
              var_accel_forearm))

# remove from test set as well
filteredTestData <-
    filteredTestData %>%
    select(-c(user_name,
              raw_timestamp_part_1,
              raw_timestamp_part_2,
              cvtd_timestamp,
              max_roll_belt:var_yaw_belt,
              var_accel_arm,
              max_picth_arm:amplitude_yaw_arm,
              max_roll_dumbbell:amplitude_pitch_dumbbell,
              var_accel_dumbbell:var_yaw_dumbbell,
              max_picth_forearm:amplitude_pitch_forearm,
              var_accel_forearm))

dim(filteredTrainData)
```

```
## [1] 19622    54
```

```
dim(filteredTestData)
```

```
## [1] 20 54
```

Preprocess

Split the training data into two set of 80% and 20%.

```
set.seed(142678)

dataIndex <- createDataPartition(filteredTrainData$classe, p = 0.8, list = FALSE)
trainSet <- filteredTrainData[dataIndex, ]
testSet <- filteredTrainData[-dataIndex, ]
```

Machine Learning

Using Random Forest and Rpart to train the model.

```
library(doParallel)
```

```
## Loading required package: foreach
```

```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

```
registerDoParallel()

modelRf <- train(classe ~ ., data = trainSet, model = "rf")
```

```
## Loading required package: randomForest
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

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

```
modelRpart <- train(classe ~ ., data = trainSet, model = "rpart")

predRf <- predict(modelRf, newdata = testSet)
predRpart <- predict(modelRpart, newdata = testSet)

C1 <- confusionMatrix(predRf, testSet$classe)
print(C1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1115    1    0    0    0
##          B    0  758    2    0    0
##          C    0    0  682    2    0
##          D    0    0    0  640    0
##          E    1    0    0    1  721
##
## Overall Statistics
##
##                Accuracy : 0.9982
##                  95% CI : (0.9963, 0.9993)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9977
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9991   0.9987   0.9971   0.9953   1.0000
## Specificity            0.9996   0.9994   0.9994   1.0000   0.9994
## Pos Pred Value         0.9991   0.9974   0.9971   1.0000   0.9972
## Neg Pred Value         0.9996   0.9997   0.9994   0.9991   1.0000
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2842   0.1932   0.1738   0.1631   0.1838
## Detection Prevalence   0.2845   0.1937   0.1744   0.1631   0.1843
## Balanced Accuracy      0.9994   0.9990   0.9982   0.9977   0.9997
```

```
C2 <- confusionMatrix(predRpart, testSet$classe)
print(C2)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1115    1    0    0    0
##          B    0  758    2    0    0
##          C    0    0  682    2    0
##          D    0    0    0  640    1
##          E    1    0    0    1  720
##
## Overall Statistics
##
##                Accuracy : 0.998
##                  95% CI : (0.996, 0.9991)
##     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            0.9991   0.9987   0.9971   0.9953   0.9986
## Specificity            0.9996   0.9994   0.9994   0.9997   0.9994
## Pos Pred Value         0.9991   0.9974   0.9971   0.9984   0.9972
## Neg Pred Value         0.9996   0.9997   0.9994   0.9991   0.9997
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2842   0.1932   0.1738   0.1631   0.1835
## Detection Prevalence   0.2845   0.1937   0.1744   0.1634   0.1840
## Balanced Accuracy      0.9994   0.9990   0.9982   0.9975   0.9990
```

Out of Sample Error

Out of sample error for Random Forest is: `{r} (1 - C1$overall[1]) * 100` . Out of sample error for Decision Tree is: `{r} (1 - C2$overall[1]) * 100` .

Predictin Result

Predicting the result using Random Forest models because of the lower out of sample error.

```
predResult <- predict(modelRf, newdata = filteredTestData)
print(predResult)
```

```
##  [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
```