Practical Machine Learning Assignment

Chuwei Zhong

10/9/2019

Load and clean data

```
#load library
library(lattice)
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.5.3
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.3
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.5.3
```

```
## 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(rpart)
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.5.3
```

```
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 3.5.3
```

```
## Loaded gbm 2.1.5
```

```
#library(rpart.plot)

#load date and convert the NA element
traincsv <- read.csv("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", na.strings=c
testcsv <- read.csv("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", na.strings=c(

#remove the columns with 90% of NA
traincsv <- traincsv[, colSums(is.na(traincsv))/nrow(traincsv) <= 0.9]
testcsv <- testcsv[, colSums(is.na(testcsv))/nrow(testcsv) <= 0.9]

#remove the columns irrelevant to our analysis
traincsv <- traincsv[,-c(1:7)]
testcsv <- testcsv[,-c(1:7)]

#Partitioning the traincsv to 80% of training set and 20% of test set to estimate the model error
traincsv_set <- createDataPartition(y=traincsv$classe, p=0.80, list=FALSE)
traincsv_training <- traincsv[traincsv_set, ]
traincsv_testing <- traincsv[-traincsv_set, ]
```

The way I do the sampling is nor so good, CV or repeatedcv are better.

First model: Decision Tree

```
# classification tree
model1 <- rpart(classe ~ ., data=traincsv_training, method="class")
prediction1 <- predict(model1, traincsv_testing, type = "class")
#rpart.plot(model1)
confusionMatrix(prediction1, traincsv_testing$classe)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction   A   B   C   D   E
##          A 987 144  44  82  17
##          B  43 463  60  53  67
##          C  38  55 515  87  81
##          D  27  59  49 359  23
##          E  21  38  16  62 533
##
## Overall Statistics
##
##                Accuracy : 0.7283
##                  95% CI : (0.7141, 0.7421)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6544
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8844   0.6100   0.7529  0.55832   0.7393
## Specificity            0.8978   0.9295   0.9194  0.95183   0.9572
## Pos Pred Value         0.7747   0.6749   0.6637  0.69439   0.7955
## Neg Pred Value         0.9513   0.9086   0.9463  0.91662   0.9422
## Prevalence             0.2845   0.1935   0.1744  0.16391   0.1838
## Detection Rate         0.2516   0.1180   0.1313  0.09151   0.1359
## Detection Prevalence   0.3248   0.1749   0.1978  0.13179   0.1708
## Balanced Accuracy      0.8911   0.7698   0.8362  0.75507   0.8482
```

Second model: Random Forest

```
model2 <- randomForest(classe ~ ., data=traincsv_training, method="class")
prediction2 <- predict(model2, traincsv_testing, type = "class")
confusionMatrix(prediction2, traincsv_testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1115    6    0    0    0
##          B    0  752    4    0    0
##          C    0    1  679    5    0
##          D    0    0    1  638    1
##          E    1    0    0    0  720
##
## Overall Statistics
##
##                Accuracy : 0.9952
##                  95% CI : (0.9924, 0.9971)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9939
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9991   0.9908   0.9927   0.9922   0.9986
## Specificity            0.9979   0.9987   0.9981   0.9994   0.9997
## Pos Pred Value         0.9946   0.9947   0.9912   0.9969   0.9986
## Neg Pred Value         0.9996   0.9978   0.9985   0.9985   0.9997
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2842   0.1917   0.1731   0.1626   0.1835
## Detection Prevalence   0.2858   0.1927   0.1746   0.1631   0.1838
## Balanced Accuracy      0.9985   0.9948   0.9954   0.9958   0.9992
```

From the outputs, Random Forest is more accurate, about 99%. So we use Random Forest for the prediction of testcsv.

Prediction

```r
prediction3 <- predict(model2, testcsv, type="class")
prediction3
```

```
##  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
```