practical machine learning course project

Lie Li

November 21, 2019

Data downloading and understanding the data

```
url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
download.file(url, destfile = "pml-training.csv")
trainingOri = read.csv("pml-training.csv",na.strings=c("NA","","#DIV/0!"))
url2 <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
download.file(url2, destfile = "pml-testing.csv")
testingOri = read.csv("pml-testing.csv")
dim(trainingOri)
```

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

```
dim(testingOri)
```

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

Data cleasing,remove N/A Value

```
trainingRemoveNA <- trainingOri[, colSums(is.na(trainingOri)) == 0]
testingRemoveNA <- testingOri[, colSums(is.na(testingOri)) == 0]
dim(trainingRemoveNA)
```

```
## [1] 19622    60
```

```
dim(testingRemoveNA)
```

```
## [1] 20 60
```

Data cleasing,remove column 1-7 which are not relavant to the modeling

```
trainingRemoveNACol   <-trainingRemoveNA[,-c(1:7)]
testingRemoveNACo   <-testingRemoveNA[,-c(1:7)]
dim(trainingRemoveNACol)
```

```
## [1] 19622    53
```

```
dim(testingRemoveNACo)
```

```
## [1] 20 53
```

Data partitioning

```
library(caret)
```

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

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

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

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

```
library(rpart)
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(gbm)
```

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

```
## Loaded gbm 2.1.5
```

```
inTrain <- createDataPartition(trainingRemoveNACol$classe, p=0.7, list=FALSE)
training <- trainingRemoveNACol[inTrain,]
testing <- trainingRemoveNACol[-inTrain,]
dim(training)
```

```
## [1] 13737    53
```

```
dim(testing)
```

```
## [1] 5885   53
```

Dicision tree model

```
modfitDT <- train(classe ~ ., data = training, method="rpart")
predictionDT <- predict(modfitDT, testing)
confusionMatrix(predictionDT, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1516  483  466  450  165
##          B   26  383   39  178  131
##          C  129  273  521  336  297
##          D    0    0    0    0    0
##          E    3    0    0    0  489
##
## Overall Statistics
##
##                Accuracy : 0.4943
##                  95% CI : (0.4815, 0.5072)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3388
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9056  0.33626  0.50780   0.0000  0.45194
## Specificity            0.6286  0.92120  0.78699   1.0000  0.99938
## Pos Pred Value         0.4922  0.50594  0.33483      NaN  0.99390
## Neg Pred Value         0.9437  0.85257  0.88334   0.8362  0.89004
## Prevalence             0.2845  0.19354  0.17434   0.1638  0.18386
## Detection Rate         0.2576  0.06508  0.08853   0.0000  0.08309
## Detection Prevalence   0.5234  0.12863  0.26440   0.0000  0.08360
## Balanced Accuracy      0.7671  0.62873  0.64740   0.5000  0.72566
```

Random forest model

```
modfitRF <- train(classe ~ ., data = training, method = "rf", ntree = 100)
predictionRF <- predict(modfitRF, testing)
confusionMatrix(predictionRF, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1669   12    0    0    0
##          B    5 1125    6    1    0
##          C    0    2 1017   12    0
##          D    0    0    3  951    4
##          E    0    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.9924
##                  95% CI : (0.9898, 0.9944)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9903
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9970   0.9877   0.9912   0.9865   0.9963
## Specificity            0.9972   0.9975   0.9971   0.9986   1.0000
## Pos Pred Value         0.9929   0.9894   0.9864   0.9927   1.0000
## Neg Pred Value         0.9988   0.9971   0.9981   0.9974   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2836   0.1912   0.1728   0.1616   0.1832
## Detection Prevalence   0.2856   0.1932   0.1752   0.1628   0.1832
## Balanced Accuracy      0.9971   0.9926   0.9942   0.9925   0.9982
```

Boosting model

```
modfitGbm <- train(classe ~ ., data = training, method = "gbm", verbose = FALSE)
modfitGbm$finalModel
```

```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 52 predictors of which 52 had non-zero influence.
```

```
predictionGbm <- predict(modfitGbm, testing)
confusionMatrix(predictionGbm, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1645   42    0    1    4
##          B   23 1066   41    5   11
##          C    3   29  976   33    8
##          D    3    2    8  920   16
##          E    0    0    1    5 1043
##
## Overall Statistics
##
##                Accuracy : 0.9601
##                  95% CI : (0.9547, 0.9649)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9495
##
##  Mcnemar's Test P-Value : 4.181e-08
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9827   0.9359   0.9513   0.9544   0.9640
## Specificity            0.9888   0.9831   0.9850   0.9941   0.9988
## Pos Pred Value         0.9722   0.9302   0.9304   0.9694   0.9943
## Neg Pred Value         0.9931   0.9846   0.9897   0.9911   0.9919
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2795   0.1811   0.1658   0.1563   0.1772
## Detection Prevalence   0.2875   0.1947   0.1782   0.1613   0.1782
## Balanced Accuracy      0.9858   0.9595   0.9681   0.9742   0.9814
```

Apply the best model RF on the test dataset

```
finalPredictionRF <- predict(modfitRF,testingRemoveNACo  )
finalPredictionRF
```

```
##  [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
```