Practical Machine Learning Course Project

Silver

11/29/2020

```
knitr::opts_chunk$set(echo = TRUE)
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.3

## Warning: package 'ggplot2' was built under R version 4.0.3
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.3
```

```
set.seed(9916)
```

Quick summary

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.

Loading the data

```
trainingRaw <- read.csv("pml-training.csv")
testingRaw <- read.csv("pml-testing.csv")

trainingEdit <- trainingRaw
testing <- testingRaw
```

Splitting and cleaning the data

Creating models

We choose to create two different machine learning algorithms and choose the more accurate one based on its accuracy on the validation set. The two choices shall be a linear discriminant analysis (LDA) approach and a random forests (RF) approach.

```
ldaModel <- train(classe ~ .,data=training,method="lda")
rfModel <- randomForest(as.factor(classe) ~ ., data = training)
```

LDA model

```
ldaTrainMat <- confusionMatrix(ldaPredTrain,as.factor(training$classe))
ldaValidMat <- confusionMatrix(ldaPredValid,as.factor(validate$classe))
ldaTrainMat
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3428  441  247  144   99
##          B   98 1823  247  102  455
##          C  343  345 1682  282  235
##          D  303  105  326 1794  260
##          E   13  134   65   90 1657
##
## Overall Statistics
##
##                Accuracy : 0.7055
##                  95% CI : (0.6981, 0.7129)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6274
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8191   0.6401   0.6552   0.7438   0.6123
## Specificity            0.9116   0.9240   0.9008   0.9192   0.9749
## Pos Pred Value         0.7864   0.6690   0.5826   0.6435   0.8458
## Neg Pred Value         0.9269   0.9145   0.9252   0.9482   0.9178
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1839
## Detection Rate         0.2329   0.1239   0.1143   0.1219   0.1126
## Detection Prevalence   0.2962   0.1851   0.1962   0.1894   0.1331
## Balanced Accuracy      0.8654   0.7821   0.7780   0.8315   0.7936
```

```
ldaValidMat
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1162  155   88   61   41
##          B   34  602  103   35  156
##          C   89  116  545   96   83
##          D  105   34  104  568   86
##          E    5   42   15   44  535
##
## Overall Statistics
##
##                Accuracy : 0.6958
##                  95% CI : (0.6827, 0.7086)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6143
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8330   0.6344   0.6374   0.7065   0.5938
## Specificity            0.9017   0.9171   0.9052   0.9198   0.9735
## Pos Pred Value         0.7711   0.6473   0.5867   0.6332   0.8346
## Neg Pred Value         0.9314   0.9127   0.9220   0.9411   0.9141
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2369   0.1228   0.1111   0.1158   0.1091
## Detection Prevalence   0.3073   0.1896   0.1894   0.1829   0.1307
## Balanced Accuracy      0.8673   0.7757   0.7713   0.8131   0.7837
```

The accuracy of the model when predicting on the training set is 0.7055306, and the out-of-sample error is 0.2944694. To control for overfitting, we predicted the model on the validation set. That gave an accuracy of 0.6957586 and an out-of-sample error of 0.3042414. This means that there was no overfitting in the model.

RF model

```
rfPredTrain <- predict(rfModel,training)
rfPredValid <- predict(rfModel,validate)
```

```
rfTrainMat <- confusionMatrix(rfPredTrain,as.factor(training$classe))
rfValidMat <- confusionMatrix(rfPredValid,as.factor(validate$classe))
rfTrainMat
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4185    0    0    0    0
##          B    0 2848    0    0    0
##          C    0    0 2567    0    0
##          D    0    0    0 2412    0
##          E    0    0    0    0 2706
##
## 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   0.1839
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1839
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1839
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```
rfValidMat
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    3    0    0    0
##          B    0  945    4    0    0
##          C    0    1  851    7    3
##          D    0    0    0  797    4
##          E    0    0    0    0  894
##
## Overall Statistics
##
##                Accuracy : 0.9955
##                  95% CI : (0.9932, 0.9972)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9943
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9958   0.9953   0.9913   0.9922
## Specificity            0.9991   0.9990   0.9973   0.9990   1.0000
## Pos Pred Value         0.9979   0.9958   0.9872   0.9950   1.0000
## Neg Pred Value         1.0000   0.9990   0.9990   0.9983   0.9983
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1927   0.1735   0.1625   0.1823
## Detection Prevalence   0.2851   0.1935   0.1758   0.1633   0.1823
## Balanced Accuracy      0.9996   0.9974   0.9963   0.9952   0.9961
```

The accuracy of the model when predicting on the training set is 1, which is expected with a random forest. To control for overfitting, we predicted the model on the validation set. That gave an accuracy of 0.9955139 and an out-of-sample error of 0.0044861. This means that there was no overfitting in the model and that it is highly accurate, thus we will choose the random forest model for our testing set as well.

Predicting results

Now we shall use our random forests model to predict the values of "classe" based on the variables provided in the testing set.

```r
finalPred <- predict(rfModel,testing)
finalPred
```

```
##  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
```