Coursera: Practical Machine Learning Course Project

Peter Gajda

The analysis is part of the Coursera Practical Machine Learning class. We use data from sport device trackers, specifically 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. The course project adresses the following task:

- Predict the manner in which they did the exercise

Loading required packages

```
require(caret)
require(randomForest)
```

Setting seed for reproducibility

```
set.seed(999)
```

Data preparation

Downloading the training and test datasets There are couple of missing values which are coded as "NA", "#DIV/0!" or "". In order to handle these data correctly we will transform these data to NA.

```
trainurl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
traindata <- read.csv(trainurl, na.strings=c("NA", "#DIV/0!", ""))

testurl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
testdata <- read.csv(testurl, na.strings=c("NA", "#DIV/0!", ""))
```

```
# Columns which contain only missing values are deleted
traindata <- traindata[,colSums(is.na(traindata)) == 0]
testdata <-testdata[,colSums(is.na(testdata)) == 0]

# The first seven columns are not necessary for our analysis, therefore we delete them
traindata <- subset(traindata, select = -c(1:7))
testdata <- subset(testdata, select = -c(1:7))
```

Preparing data sets for cross validation

In order to perform cross validation, we need to split the training dataset into two data sets. 80 % of the the training data is partioned to the training_subset variable. The remaining 20 % will be set to our

validation_subset variable, which enables later cross validation. We use createDataPartition from the caret package and create two data matrices.

```
DataPartitions <- createDataPartition(traindata$classe, p = 0.80, list = FALSE)

training_subset <- traindata[DataPartitions, ]
validation_subset <- traindata[-DataPartitions, ]
```

Random Forest

We will use a random forest decision tree for our model training, due to its accuracy. The algorithm finds the influencing variables by averaging the results of different decision trees. We use the randomForest function from the random forest package. The size of dataset is no limitation for us using this method.

```
rfclass <- randomForest(classe ~. , data=training_subset, method="class")
rfclass
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training_subset, method = "class")
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.39%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 4462    2    0    0    0 0.0004480287
## B    9 3024    5    0    0 0.0046082949
## C    0   17 2719    2    0 0.0069393718
## D    0    0   23 2549    1 0.0093276331
## E    0    0    1    2 2883 0.0010395010
```

Prediction

We use our model from the training_subset and test it against the validation_subset. We use confusionMatrix from the caret package in order to compare the predicted values for the validation data against the actual data from validation_subset.

```
prediction <- predict(rfclass, validation_subset, type = "class")
confusionMatrix(prediction, validation_subset$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1115    4    0    0    0
##          B    0  752    2    0    0
##          C    0    3  682    2    0
```

```
##          D    0    0    0  640    0
##          E    1    0    0    1  721
##
## Overall Statistics
##
##                Accuracy : 0.9967
##                  95% CI : (0.9943, 0.9982)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9958
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9991   0.9908   0.9971   0.9953   1.0000
## Specificity            0.9986   0.9994   0.9985   1.0000   0.9994
## Pos Pred Value         0.9964   0.9973   0.9927   1.0000   0.9972
## Neg Pred Value         0.9996   0.9978   0.9994   0.9991   1.0000
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2842   0.1917   0.1738   0.1631   0.1838
## Detection Prevalence   0.2852   0.1922   0.1751   0.1631   0.1843
## Balanced Accuracy      0.9988   0.9951   0.9978   0.9977   0.9997
```

The accuracy is 0.9967, hence the Overall-Out-Of-Sample Error is 0.0033 This implies that only very few data of the test sample will be classified for the wrong variable.

Applying the model to the testdata

```
submission <- predict(rfclass, testdata, type="class")
submission
```

```
##  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
```