Prediction-Assignment-Writeup

Intro

The goal of this project is to predict the manner in which the subjects did the exercise. This is the "classe" variable in the training set.

The "classe" are: - exactly according to the specification (Class A) - throwing the elbows to the front (Class B) - lifting the dumbbell only halfway (Class C) - lowering the dumbbell only halfway (Class D) - throwing the hips to the front (Class E)

Load Libraries

```
library(caret)
```

```
## Lade nötiges Paket: ggplot2

## Lade nötiges Paket: lattice
```

```
library(randomForest)
```

```
## randomForest 4.7-1

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

##
## Attache Paket: 'randomForest'

## Das folgende Objekt ist maskiert 'package:ggplot2':
##
##     margin
```

```
library(rpart)
library(rpart.plot)
```

Download Files

```
url_train = 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
url_test = 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'

download.file(url_train, 'train.csv')
download.file(url_test, 'test.csv')
```

Load Files

```
train_set = read.csv('train.csv', na.strings=c("NA","#DIV/0!", ""))
test_set = read.csv('test.csv', na.strings=c("NA","#DIV/0!", ""))
```

Set seed

```
set.seed(1909)
```

Data Cleanup, convert classe to factor and Subset

```
train_set <- train_set[,colSums(is.na(train_set)) == 0]
test_set <- test_set[,colSums(is.na(test_set)) == 0]

train_set$classe <- as.factor(train_set$classe)

train_set <- train_set[,-c(1:7)]
test_set <- test_set[,-c(1:7)]
```

Splitting Data for cross-validation

```
samples_cv <- createDataPartition(y=train_set$classe, p=0.75, list=FALSE)
train_cv <- train_set[samples_cv, ]
test_cv <- train_set[-samples_cv, ]
```

Prediction models

random forest

```
model_rf <- randomForest(classe ~ ., data=train_cv, method="class")
predict_rf <- predict(model_rf, test_cv, type = "class")
```

```
confusionMatrix(predict_rf, test_cv$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1392    1    0    0    0
##          B    3  947    4    0    0
##          C    0    1  849   13    0
##          D    0    0    2  791    1
##          E    0    0    0    0  900
```

```
##
## Overall Statistics
##
##                Accuracy : 0.9949
##                  95% CI : (0.9925, 0.9967)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9936
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9978   0.9979   0.9930   0.9838   0.9989
## Specificity            0.9997   0.9982   0.9965   0.9993   1.0000
## Pos Pred Value         0.9993   0.9927   0.9838   0.9962   1.0000
## Neg Pred Value         0.9991   0.9995   0.9985   0.9968   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2838   0.1931   0.1731   0.1613   0.1835
## Detection Prevalence   0.2841   0.1945   0.1760   0.1619   0.1835
## Balanced Accuracy      0.9988   0.9981   0.9948   0.9915   0.9994
```

Decision tree

```
model_dt <- rpart(classe ~ ., data=train_cv, method="class")
predict_dt <- predict(model_dt, test_cv, type = "class")
```

```
confusionMatrix(predict_dt, test_cv$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1271  129   12   44   11
##          B   48  581   79   95  110
##          C   31  138  688  128  125
##          D   18   70   53  494   50
##          E   27   31   23   43  605
##
## Overall Statistics
##
##                Accuracy : 0.742
##                  95% CI : (0.7296, 0.7542)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6731
##
##  Mcnemar's Test P-Value : < 2.2e-16
```

```
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9111   0.6122   0.8047   0.6144   0.6715
## Specificity            0.9441   0.9161   0.8958   0.9534   0.9690
## Pos Pred Value         0.8664   0.6364   0.6198   0.7212   0.8299
## Neg Pred Value         0.9639   0.9078   0.9560   0.9265   0.9291
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2592   0.1185   0.1403   0.1007   0.1234
## Detection Prevalence   0.2991   0.1862   0.2263   0.1397   0.1487
## Balanced Accuracy      0.9276   0.7641   0.8502   0.7839   0.8202
```

Summary

The comparison of the confusion matrices shows that the RandomForest model performs better than the DecisionTree model and should therefore be selected.

The RandmForest model has an accuracy of 0.9949 with a 95% confidence interval of (0.9925, 0.9967). The DecisionTree model has an accuracy of 0.742 with a 95% confidence interval of (0.7296, 0.7542).

Expected out-of-sample error

The expected out-of-sample error is estimated at 0.005. The expected out-of-sample error is calculated via (1 - accuracy) the estimate of the cross-validation set. With an accuracy of 0.9949 in the RandomForest model on the cross-validation set, we can assume that only a few of the 20 cases in the test data set are misclassified.

Submission

```
predict_sub <- predict(model_rf, test_set, type="class")
predict_sub
```

```
##  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
```

```
write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
write_files(predict_sub)
```