Practical Machine Learning Course Project

Introduction

There are many opportunities to measure human activities due to the availability of low cost accelerometers. We will examine whether we can determine the weight lifting form using the accelerometer data collected.

Exploring and preparing the data

```
# set working directory
setwd("D:/CURSOS/DATA SCIENCE/8-Practical Machine Learning/Course
project")

library(caret)
```

```
## Loading required package: lattice
## Loading required package: ggplot2
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:survival':
##
##     cluster
```

```

# Change to data directory
setwd("./datasets/")

# Load datasets
trainingDB <- read.csv("pml-training.csv", header = TRUE, sep =
",", stringsAsFactors = FALSE)
testingDB <- read.csv("pml-testing.csv", header = TRUE, sep = ",",
stringsAsFactors = FALSE)
```

We choose random selection without replacement to split the data set into a training set (70%) and a cross validation set (30%). Set seed for reproducibility pourposes

```
set.seed(1535)
trainingIndex <- createDataPartition(trainingDB$classe, list =
FALSE, p = 0.7)
training = trainingDB[trainingIndex, ]
testing = trainingDB[-trainingIndex, ]
```

Remove indicators with near zero variance.

```
nzv <- nearZeroVar(training)
training <- training[-nzv]
testing <- testing[-nzv]
testingDB <- testingDB[-nzv]
```

We filter columns to include only numeric features and outcome.

```
num_features = which(lapply(training, class) %in% c("numeric"))
```

We impute missing values in our training data.

```
Model <- preProcess(training[, num_features], method =
c("knnImpute"))
pretrain <- cbind(training$classe, predict(Model, training[,
num_features]))
pretest <- cbind(testing$classe, predict(Model, testing[,
num_features]))
prtesting <- predict(Model, testingDB[, num_features])

# Fix label on classe
names(pretrain)[1] <- "classe"
names(pretest)[1] <- "classe"
```

Random Forest model

We build a random forest model using the numerical variables provided.

```
library(randomForest)
```

```
## randomForest 4.6-7
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:Hmisc':
##
##     combine
```

```
modelRF <- randomForest(classe ~ ., pretrain, ntree = 500, mtry =
32)
```

Cross Validation

In-sample accuracy

```
train_pred <- predict(modelRF, pretrain)
print(confusionMatrix(train_pred, pretrain$classe))
```

```
## Loading required package: class
##
## Attaching package: 'e1071'
##
## The following object is masked from 'package:Hmisc':
##
##     impute
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3906    0    0    0    0
##          B    0 2658    0    0    0
##          C    0    0 2396    0    0
##          D    0    0    0 2252    0
##          E    0    0    0    0 2525
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (1, 1)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class:
E
## Sensitivity            1.000    1.000    1.000    1.000
1.000
## Specificity            1.000    1.000    1.000    1.000
1.000
## Pos Pred Value         1.000    1.000    1.000    1.000
1.000
## Neg Pred Value         1.000    1.000    1.000    1.000
1.000
## Prevalence             0.284    0.193    0.174    0.164
0.184
## Detection Rate         0.284    0.193    0.174    0.164
0.184
## Detection Prevalence   0.284    0.193    0.174    0.164
0.184
## Balanced Accuracy      1.000    1.000    1.000    1.000
1.000
```

The in-sample accuracy is 100%. This indicates that the model does not suffer from bias.

Out-of-sample accuracy

```
test_pred <- predict(modelRF, pretest)
```

```
print(confusionMatrix(test_pred, pretest$classe))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1670   10    1    0    0
##          B    2 1122   10    3    3
##          C    0    4  999    6    1
##          D    0    3   15  954    4
##          E    2    0    1    1 1074
##
## Overall Statistics
##
##                Accuracy : 0.989
##                  95% CI : (0.986, 0.991)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.986
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class:
E
## Sensitivity            0.998    0.985    0.974    0.990
0.993
## Specificity            0.997    0.996    0.998    0.996
0.999
## Pos Pred Value         0.993    0.984    0.989    0.977
0.996
## Neg Pred Value         0.999    0.996    0.994    0.998
0.998
## Prevalence             0.284    0.194    0.174    0.164
0.184
## Detection Rate         0.284    0.191    0.170    0.162
0.182
## Detection Prevalence   0.286    0.194    0.172    0.166
0.183
## Balanced Accuracy      0.997    0.991    0.986    0.993
0.996
```

The cross validation accuracy is greater than 99%. Based on the lower bound of the confidence interval we would expect to achieve a 98.7% classification accuracy on new data provided.

Evaluating model performance in the test set

We apply this model to the test data provided and we get 100% classification accuracy on the twenty test observations.

```
obs <- predict(modelRF, prtesting)
obs
```

```
##  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
```

Conclusions

Our model provides good accuracy to predict the twenty test cases. We are able to provide very good prediction of weight lifting style as measured by accelerometers.