Practical Machine Learning: Course project

The goal of the project

The description of the assignment contains the following information on the dataset:

> 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.

The goal is

> to predict the manner in which they did the exercise.

The training of the model

In the following, I describe the steps concerning the training of a predictive model.

Read the data

First, the `.csv` file contain the training data is read into R. Here, unavailable values are set as `NA`.

```
rawData <- read.csv("./data/pml-training.csv", na.strings = c("NA", ""))
```

Reduce the dataset

In the next step, I check the proportion of missing values (`NA` s) in the columns.

```
propNAs <- colMeans(is.na(rawData))
table(propNAs)
```

```
## propNAs
##                0 0.979308938946081
##               60               100
```

There are 100 columns in which almost all values (97.93%) are missing. If a column contains a large number of `NA` s, it will not be of great use for training the model. Hence, these columns will be removed. Only the columns without any `NA` s will be kept.

```
# index of columns with NA values
idx <- !propNAs
# check
sum(idx)
```

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

```
# remove these columns
rawDataReduced <- rawData[idx]
# check
ncol(rawDataReduced)
```

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

There are further unnecessary columns that can be removed. The column `X` contains the row numbers. The column `user_name` contains the name of the user. Of course, these variables cannot predictors for the type of exercise.

Furthermore, the three columns containing time stamps (`raw_timestamp_part_1` , `raw_timestamp_part_2` , and `cvtd_timestamp`) will not be used.

The factors `new_window` and `num_window` are not related to sensor data. They will be removed too.

```
# find columns not containing sensor measurement data
idx <- grep("^X$|user_name|timestamp|window", names(rawDataReduced))
# check
length(idx)
```

```
## [1] 7
```

```
# remove columns
rawDataReduced2 <- rawDataReduced[-idx]
```

Preparing the data for training

Now, the dataset contains one outcome column (`classe`) and 59 feature columns. The function `createDataPartition` of the `caret` package is used to split the data into a training and a cross-validation data set. Here, 70% of the data goes into the training set.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.1
```

```
## Warning: package 'lattice' was built under R version 3.4.1
```

```
## Warning: package 'ggplot2' was built under R version 3.4.1
```

```
inTrain <- createDataPartition(y = rawDataReduced2$classe, p = 0.7, list = FALSE)
```

The index `inTrain` is used to split the data.

```
training <- rawDataReduced2[inTrain, ]
# the number of columns on the training set
nrow(training)
```

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

```
crossval <- rawDataReduced2[-inTrain, ]
# the number of rows in the cross-validation set
nrow(crossval)
```

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

Train a model

I used the *random-forest* technique to generate a predictive model. In sum, 10 models were trained. I played around with the parameters passed to `trControl` and specified different models with bootstrapping (`method = "boot"`) and cross-validation (`method = "cv"`).

It took more than one day to train all models. Afterwards I tested their performance on the cross-validation dataset. It turned out that all models showed a good performance (because their accuracy was above 99%) though their training times were quite different.

Due to the similar performance, I will present the model with the shortest training time.

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.1
```

```
trControl <- trainControl(method = "cv", number = 2)
modFit <- train(classe ~ ., data = training, method = "rf", prox = TRUE, trControl = trContro
l)
```

Evaluate the model (out-of-sample error)

First, the final model is used to predict the outcome in the cross-validation dataset.

```
pred <- predict(modFit, newdata = crossval)
```

Second, the function `confusionMatrix` is used to calculate the accuracy of the prediction.

```
coMa <- confusionMatrix(pred, reference = crossval$classe)
acc <- coMa$overall["Accuracy"]
acc
```

```
##  Accuracy
## 0.9942226
```

The accuracy of the prediction is 99.42%. Hence, the *out-of-sample error* is 0.58%.

Variable importance

The five most important variables in the model and their relative importance values are:

```
vi <- varImp(modFit)$importance
vi[head(order(unlist(vi), decreasing = TRUE), 5L), , drop = FALSE]
```

```
##                      Overall
## roll_belt         100.00000
## pitch_forearm      58.90347
## yaw_belt           51.37440
## pitch_belt         43.02363
## magnet_dumbbell_y  42.32896
```

The source of the data

The assignment is based on data of weight lifting exercises. It has been published:

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises (http://groupware.les.inf.puc-rio.br/har#ixzz34irPKNuZ). *Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13)*. Stuttgart, Germany: ACM SIGCHI, 2013.