Course Project: Prediction Assignment (Practical Machine Learning)

Instructions

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. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Data Processing

```
library(tidyverse)
library(caret)
library(randomForest)
train_raw <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")
test_raw <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")
dim(train_raw)
```

```
## [1] 19622   160
```

```
dim(test_raw)
```

```
## [1]  20 160
```

Since the data set has a lot of columns (160) and the test data set contains 19,622 observations, I decided to delete all columns with missing values, as this limit the prediction opportunities.

Therefore the train and test data is reduced by all columns that contain missing values in one of the two data sets. "classe" is the prediction variable and needs to be converted to factor.

```
train_data <- train_raw[,colSums(is.na(train_raw)) == 0 & colSums(is.na(test_raw)) == 0]
test_data <- test_raw[,colSums(is.na(test_raw)) == 0 & colSums(is.na(train_raw)) == 0]
train_data$classe <- factor(train_data$classe)
dim(train_data)
```

```
## [1] 19622    60
```

```
dim(test_data)
```

```
## [1] 20 60
```

In the next step the train data will be split in a train- and a validation portion, which allows for testing the final model on data that was not used for prediction, befonre apllying the model to the final test set.

```
set.seed(2021)
inTrain <- createDataPartition(y=train_data$classe, p=0.7, list = FALSE)
training <- train_data[inTrain,]
validation <- train_data[-inTrain,]
```

By inspecting the variables, there are some which seem not be useful as predictor: user_name and some variables with time information. There might me even more to exclude, but I decided to just exclude those obviously useless variables for the first try.

```
relevant <- grepl("time", names(training), fixed = TRUE)
relevant[1] <- TRUE
train <- training[,-relevant]
val <- validation[,-relevant]
test <- test_data[,-relevant]
dim(train)
```

```
## [1] 13737    59
```

Model fitting

For the model fitting, I decided to use the Random Forest method: It can handle binary features, categorical features, and numerical features. There is very little pre-processing that needs to be done. The data does not need to be re-scaled or transformed.

The dimensions of the train data indicate the calculation will not take too long. After fitting the model, I will test the accuracy on the train set by comparing the model outcome to the actuall classee values in the train set.

```
modFit <- randomForest(classe~., data=train)
results <- predict(modFit, train)
accuracy <- sum(results == training$classe)/length(results)
print(paste0(as.character(accuracy*100), "%"))
```

```
## [1] "100%"
```

On the train set the accuracy reaches 100%. ALl observations are categorized correctly. In the next step the model will be tested on the validation set.

```
results_validation <- predict(modFit, val)
accuracy <- sum(results_validation == validation$classe)/length(results_validation)
print(paste0(as.character(round(accuracy,4)*100), "%"))
```

```
## [1] "99.95%"
```

```
table(results_validation, val$classe)
```

```
##
## results_validation    A    B    C    D    E
##                  A 1674    0    0    0    0
##                  B    0 1139    1    0    0
##                  C    0    0 1025    2    0
##                  D    0    0    0  962    0
##                  E    0    0    0    0 1082
```

Accuracy on the validation set is 99.95% (only 3 observations were predicted worngly), which is an accurate result. Therefore I expect a similar accuracy on the test set (~99%), indicating a low out of sample error.

```
results_test <- predict(modFit, test)
results_test
```

```
##  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
```

The prediction quiz conformed that all 20 test observations are predicted correctly with the model.