Peer-graded Assignment: Practical Machine Learning

Background

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.

The aim of this exercise is to predict the way participants conducted physical exercises based on a row of predictors. The training data contains the variable "classe", which describes how participants conducted the exercises. Possible levels for this variable are A,B,C,D and E. The following file contains the steps of data preprocessing, data exploration and model building using random forest prediction. Finally, the model is cross-validated in an independent test.

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source:

http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har.

Load training data

The dataset contains quite a lot of empty fields and fields containing "#DIV/0!". Therefore, it is specified that empty strings as well as fields containing "#DIV/0!" are seen as NA:

```r
set.seed(1234) #Set random seed

data <- read.csv("pml-training.csv", na.strings =c("", "NA", "#DIV/0!")) #Load dataset
```

Preprocessing

Useless variables that mainly contain NAs are removed Also, the first 7 columns are removed as they do not represent meaningful predictors. This leaves 53 variables in the dataset. Furthermore, the target variable "classe" is transformed from a character to a factor variable:

```r
data <- data[, colSums(is.na(data)) == 0] #Remove columns that contain NAs

data <- data[, -c(1:7)] #Remove first 7 columns

data$classe <- as.factor(data$classe) #Transform variable classe to factor
```

The dataset is split into a training and validation set (70% training, 30% validation) for cross-validation purposes:

```r
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```r
inTrain <- createDataPartition(y=data$classe,
                               p=0.7,list=F)

training <- data[inTrain,]
validation <- data[-inTrain,]
```

Exploratory data analyses

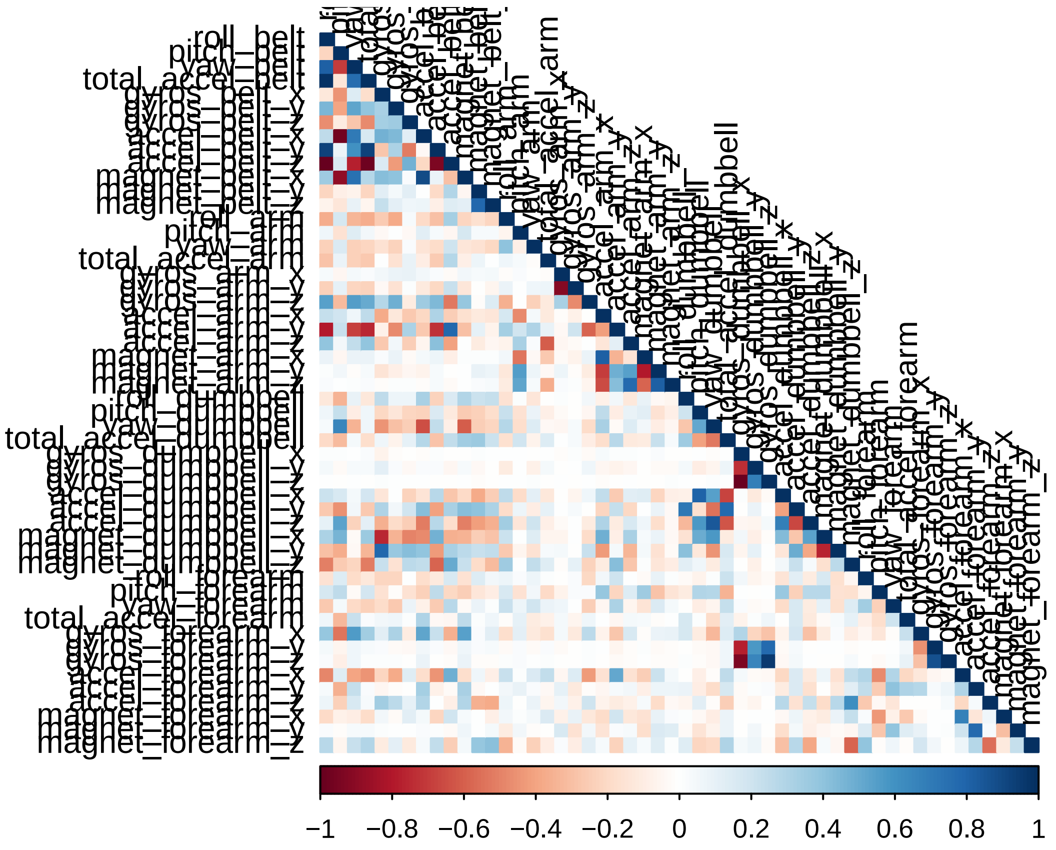
To explore the correlations between predictors, a correlation matrix is plotted, showing the correlation between all predictors:

```r
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.0.3
```

```
## corrplot 0.84 loaded
```

```r
corrMat <- cor(training[,-53])
corrplot(corrMat, method= "color", type= "lower", tl.col=rgb(0,0,0))
```



Fitting the model

A random-forest model is fitted using the train function from the caret package. The model specifications are fine-tuned to use 3-fold cross-validation, mainly for performance reasons. The final fit of the model shows high accuracy.

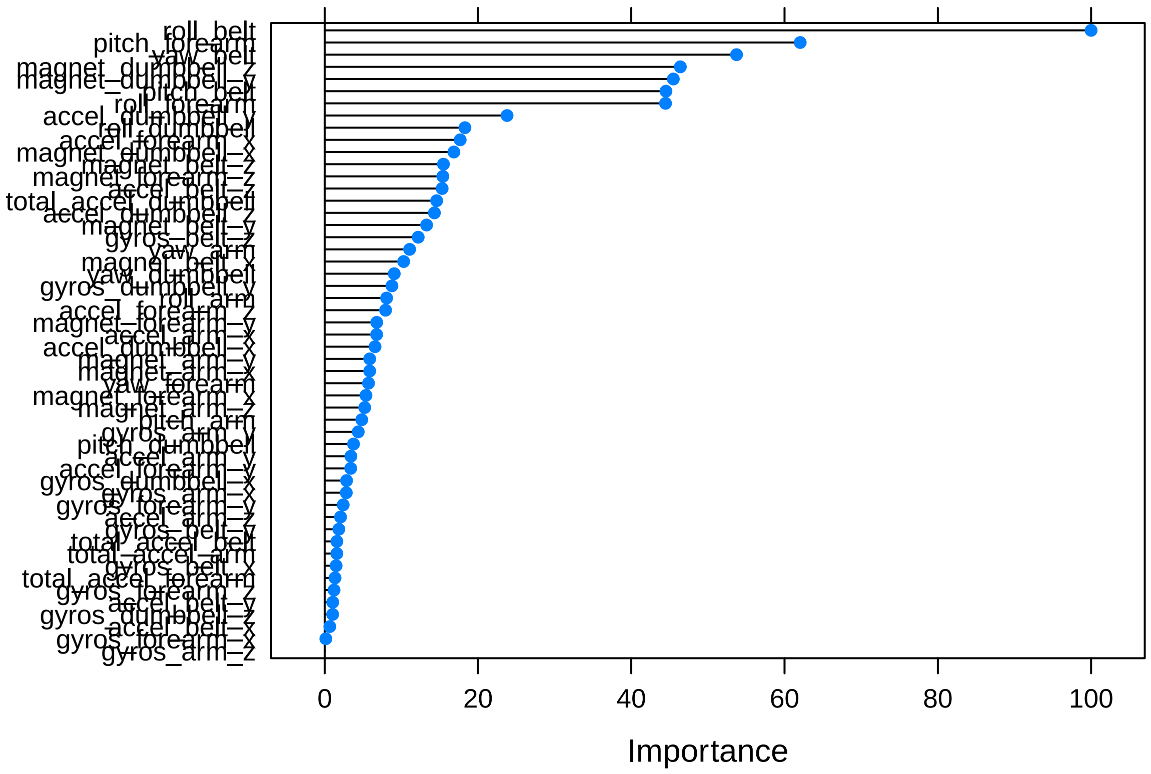
```r
control <- trainControl(method="cv",number=3,verboseIter = FALSE)
modFit <- train(classe ~ ., data=training, method="rf", trControl= control)

modFit #Print model fit
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (3 fold)
## Summary of sample sizes: 9158, 9159, 9157
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9883527  0.9852649
##   27    0.9887168  0.9857269
##   52    0.9816556  0.9767918
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

To get an overview of the importance of the predictors, the importance of variables is plotted:

```r
plot(varImp(modFit))
```



The model is then used on the cross-validation set in order to determine the accuracy in predicting the target variable in an independent validation set. In addition, a confusion matrix is printed to examine the accuracy of the model in the validation sampl:

```r
pred <- predict(modFit, validation) #Predict the target variable in the validation set

confusionMatrix(pred,validation$classe) #Print a confusion matrix of the results
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    5    0    0    0
##          B    0 1130   12    0    0
##          C    1    4 1012    9    0
##          D    0    0    2  954    0
##          E    0    0    0    1 1082
##
## Overall Statistics
##
##                Accuracy : 0.9942
##                  95% CI : (0.9919, 0.996)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9927
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9921   0.9864   0.9896   1.0000
## Specificity            0.9988   0.9975   0.9971   0.9996   0.9998
## Pos Pred Value         0.9970   0.9895   0.9864   0.9979   0.9991
## Neg Pred Value         0.9998   0.9981   0.9971   0.9980   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1920   0.1720   0.1621   0.1839
## Detection Prevalence   0.2851   0.1941   0.1743   0.1624   0.1840
## Balanced Accuracy      0.9991   0.9948   0.9917   0.9946   0.9999
```

The output shows that the accuracy is very high, 99.42%.

Using the model in the test set

For this purpose, the test set data containing 20 observations is loaded and preprocessed just like the training set before. The only difference is that "classe" is not transformed to a factor variable, because the testing dataset does not contain this variable:

```r
test <- read.csv("pml-testing.csv", na.strings =c("", "NA", "#DIV/0!"))
test <- test[, colSums(is.na(test)) == 0]
test <- test[, -c(1:7)]
```

Finally, the target variable "classe" is predicted in the training set using the trained model and the predicted classess for each observation are printed.

```r
predT <- predict(modFit, test)

predT
```

```
##  [1] 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
```