Practical Machine Learning

Sebastian Fischer

22 Mai 2018

1. 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. 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).

2. Data import and preparation

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://groupware.les.inf.puc-rio.br/har. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

```r
# Download the data
training <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"))
testing <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"))

# create subset for training data for validation

partTrain <- createDataPartition(training$classe, p=0.7, list=F)
TrainSet <- training[partTrain, ]
TestSet <- training[-partTrain, ]
dim(TrainSet)
```

```
## [1] 13737   160
```

```r
dim(TestSet)
```

```
## [1] 5885  160
```

The train and the test subsets consist of 160 variables. Lots of the variables contain NAs or low information content that have to be cleared. Different steps of clearing were performed.

Remove NAs

Several variables are empty and contain only NAs which can be removed.

```
# Remove variables with more than 50% NAs

NAs <- sapply(TrainSet, function(x) mean(is.na(x))) > 0.5
TrainSet <- TrainSet[, NAs==FALSE]
TestSet  <- TestSet[, NAs==FALSE]
dim(TrainSet)
```

```
## [1] 13737    93
```

```
dim(TestSet)
```

```
## [1] 5885   93
```

The number of variables could be decreased to 93.

Remove Variables with Nearly Zero Variance

Variables with nearly no variance are not helpful in prediction and have to be eliminated.

```
# Remove NZV
NZV <- nearZeroVar(TrainSet)
TrainSet <- TrainSet[, -NZV]
TestSet <- TestSet[, -NZV]
dim(TrainSet)
```

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

```
dim(TestSet)
```

```
## [1] 5885   59
```

The number of variables could be reduced furthermore to 59.

Remove variables only required for the identification.

The first 5 rows contain variables that are not useful for the prediction.

```
# Remove identifier
TrainSet <- TrainSet[, -(1:5)]
TestSet  <- TestSet[, -(1:5)]
dim(TrainSet)
```

```
## [1] 13737    54
```

```
dim(TestSet)
```

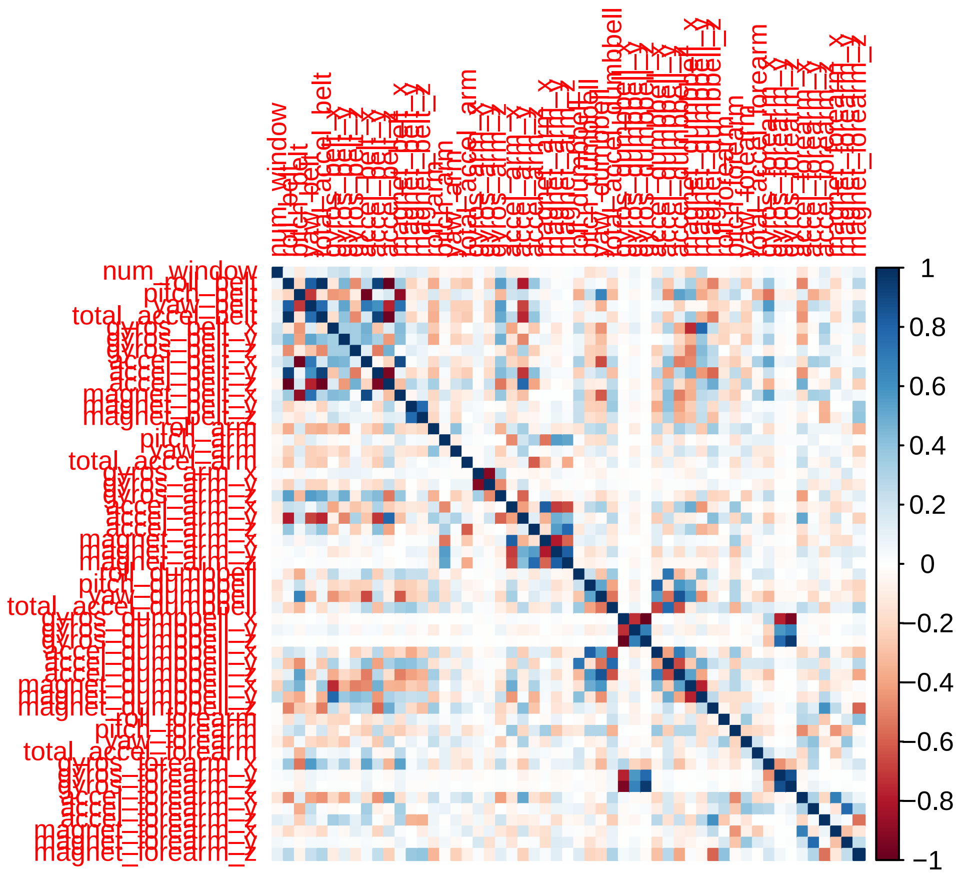
```
## [1] 5885   54
```

54 informative variables remain for the analysis.

Correlation analysis

Next, it is tested if there are strong correlations between the parameters.

```r
Matrix <- cor(TrainSet[, -54])
corrplot(Matrix, method = "color", tl.cex = 0.8)
```



As visible in the plot there are only very few correlations between the variables and it can be assumed that they are independant.

Building the model

Random forest was selceted for creating the model on the train data set.

```r
# Building the model
set.seed(123)
parameters <- trainControl(method="oob", number=5, verboseIter=FALSE)
modFitRandForest <- train(classe ~ ., data=TrainSet, method="rf",
                          trControl=parameters)
modFitRandForest$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.19%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    1    0    0    1 0.0005120328
## B    6 2650    1    1    0 0.0030097818
## C    0    5 2391    0    0 0.0020868114
## D    0    0    7 2245    0 0.0031083481
## E    0    0    0    4 2521 0.0015841584
```

The estimated OOB error rate is with 0.19% low.

Next, the model is applied to the validation set.

```
validation <- predict(modFitRandForest , newdata=TestSet)
confMatrixValidation <- confusionMatrix(validation,TestSet$classe)
confMatrixValidation
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1133    4    0    0
##          C    0    1 1022    8    0
##          D    0    0    0  956    4
##          E    0    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.9963
##                  95% CI : (0.9943, 0.9977)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9953
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9961   0.9917   0.9963
## Specificity            0.9988   0.9992   0.9981   0.9992   1.0000
## Pos Pred Value         0.9970   0.9965   0.9913   0.9958   1.0000
## Neg Pred Value         1.0000   0.9987   0.9992   0.9984   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1925   0.1737   0.1624   0.1832
## Detection Prevalence   0.2853   0.1932   0.1752   0.1631   0.1832
## Balanced Accuracy      0.9994   0.9969   0.9971   0.9954   0.9982
```

The accurancy is 99.63%. That is a very good result and the model can be used for the 20 test cases.

Validation of final data set

```
final <- predict(modFitRandForest, newdata=testing)
final
```

```
##  [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
```