Coursera Practical Machine Learning Assignment

Gerrit Timmerhaus

April 27, 2016

Introduction

Small portable devices like Jawbone Up, Nike FuelBand, and Fitbit allow to record large amounts of personal activity data relatively inexpensively. In this assignment, sensor data from correct and incorrect movements of dumbbell lifts were used to fit and validate a machine learning model. The data came from the Human Activity Recognition Project. The projet description and further information can be found on the website http://groupware.les.inf.puc-rio.br/har.

To collect the data, six young and healthy participants were asked to perform one set of 10 repetitions of the unilateral dumbbell biceps curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E). The participants were wearing sensors on belt, forearm, arm and the dumbbell. These data were used to fit a random forest model. The resulting model was able to predict the class with very high precision (>99.8% accuracy).

Exploratory Data Analysis and Filtering

The data was downloaded directly from the internet. Missing values and "#DIV/0!"-values were replaced by NA. The caret library was loaded and the seet was set to ensure consistent results.

```
training <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
                     na.strings = c("", "NA", "#DIV/0!"), stringsAsFactors = F)
testing  <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",
                     na.strings = c("", "NA", "#DIV/0!"), stringsAsFactors = F)
library(caret)
```

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

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

```
set.seed(1)
```

The functions *dim* and *table* were used to get a first overview over the data:

```
dim(training)
```

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

```
table(training$classe, training$user_name)
```

```
##
##     adelmo carlitos charles eurico jeremy pedro
##   A   1165      834     899    865   1177   640
##   B    776      690     745    592    489   505
##   C    750      493     539    489    652   499
##   D    515      486     642    582    522   469
##   E    686      609     711    542    562   497
```

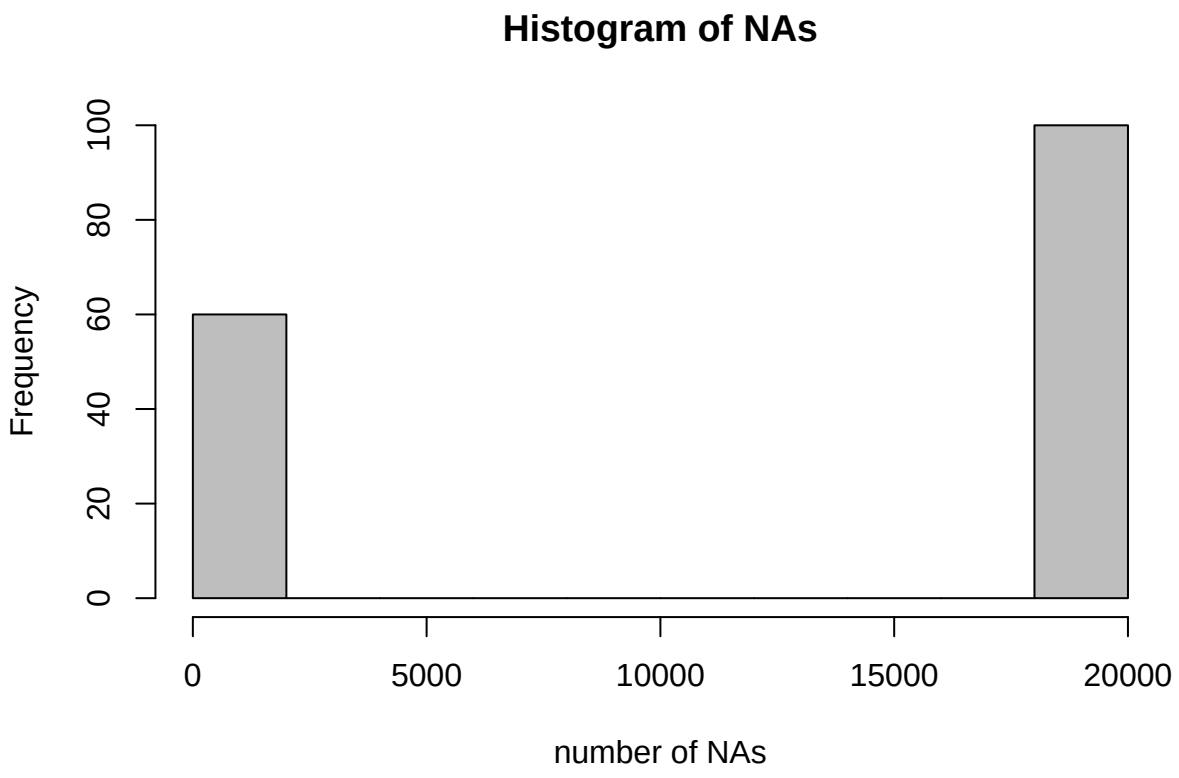
```
#table(testing$problem_id, testing$user_name)
which(names(training) != names(testing))
```

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

Testing contained 19622 observations and testing contained 20. Both data sets had the same number of columns (160) and the same column names except the last column. This column contained the class (*classe*) identifier for the training set (A-E) and for testing the *problem_ID*, a number from 1 to 20. The training data set was used to fit and validat the prediction model in the following sections. The testing data set was used for the Coursera Prediction Quiz in the last section.

A large number of columns contain mostly NAs; thus, NAs were counted in each column and plotted in a histogram:

```
nas <- apply(training, 2, function(x) sum(is.na(x)))
hist(nas, xlab="number of NAs", col="grey", main="Histogram of NAs")
```

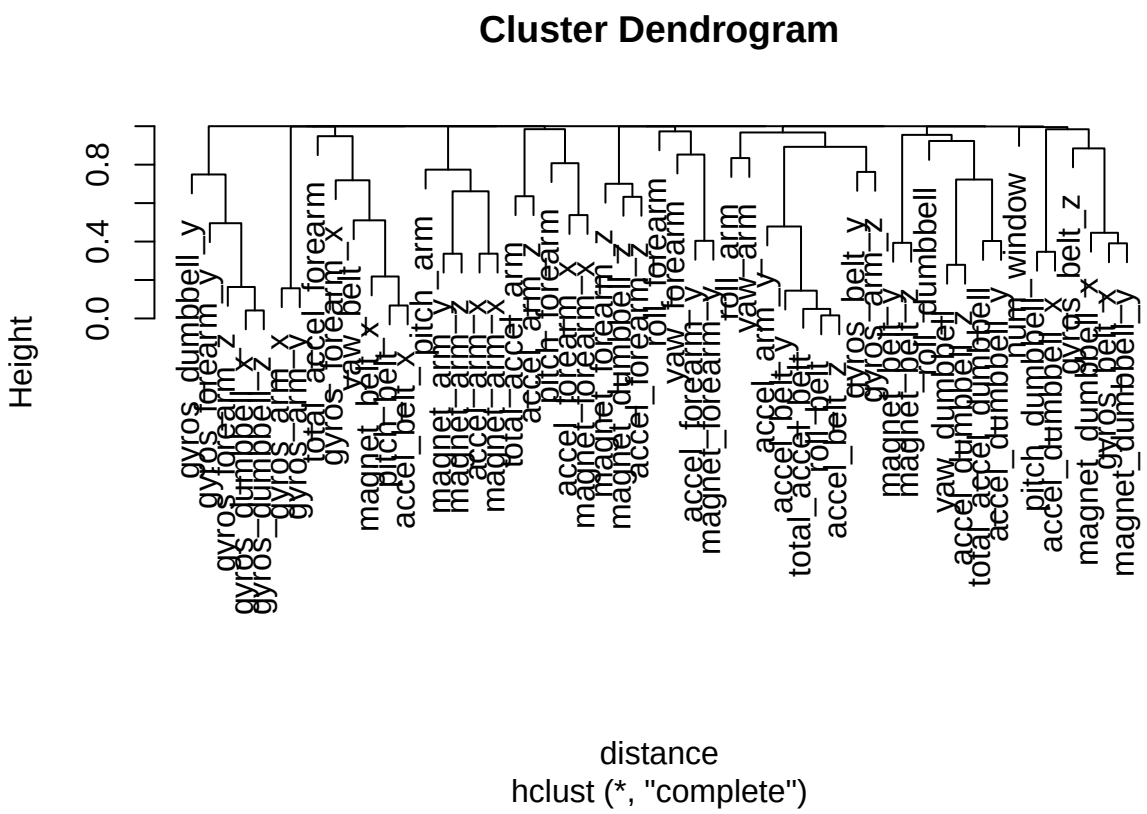


The histogram showed that 100 columns contained very high proportions of NAs and 60 columns contained almost no NAs. The 100 NA-rich columns were removed from the data sets. In addition, the first 6 columns were removed from the data set (containing date, time, participant name etc.), because they were not relevant for the further analysis.

```
training <- training[,nas<1000]
testing <- testing[,nas<1000]
training <- training[,-1:-6]
testing <- testing[,-1:-6]
```

The correlations (R^2) between the remaining 54 columns were checked in a cluster dendrogram. The distance matrix was calculated from the formula $1 - R^2$:

```
#cluster according to correaltion:
distance <- as.dist(1-(cor(training[,-54])^2))
plot(hclust(distance))
```



This showed that many of the variables were strongly correlated to each other. Thus, the number of predictors can probably be reduced in the model building.

The data set was split into an actual training set (60%) and a validation set (40%). In addition, a smaller data set (5%) was created, which was only used to identify the most important predictors.

```
temp <- createDataPartition(y=training$classe, p=0.60, list=FALSE)
training1  <- training[temp,]
validation  <- training[-temp,]
training_short <- training[createDataPartition(y=training$classe, p=0.05, list=FALSE),]
```

Modeling

Data Reduction

The random forest algorithm from the *carot* package was used to calculate a first model to discriminate the *classe* variable in the short training set by using all available 53 columns.

```
rf_model<-train(classe~., data=training_short, method="rf")
```

```
## Loading required package: randomForest
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

This small data set took several minutes to calculate; thus, the data was reduced for the following models. The function *varImp* was used on the first model to identify the relative importance of the variables:

```
varImp(rf_model)
```

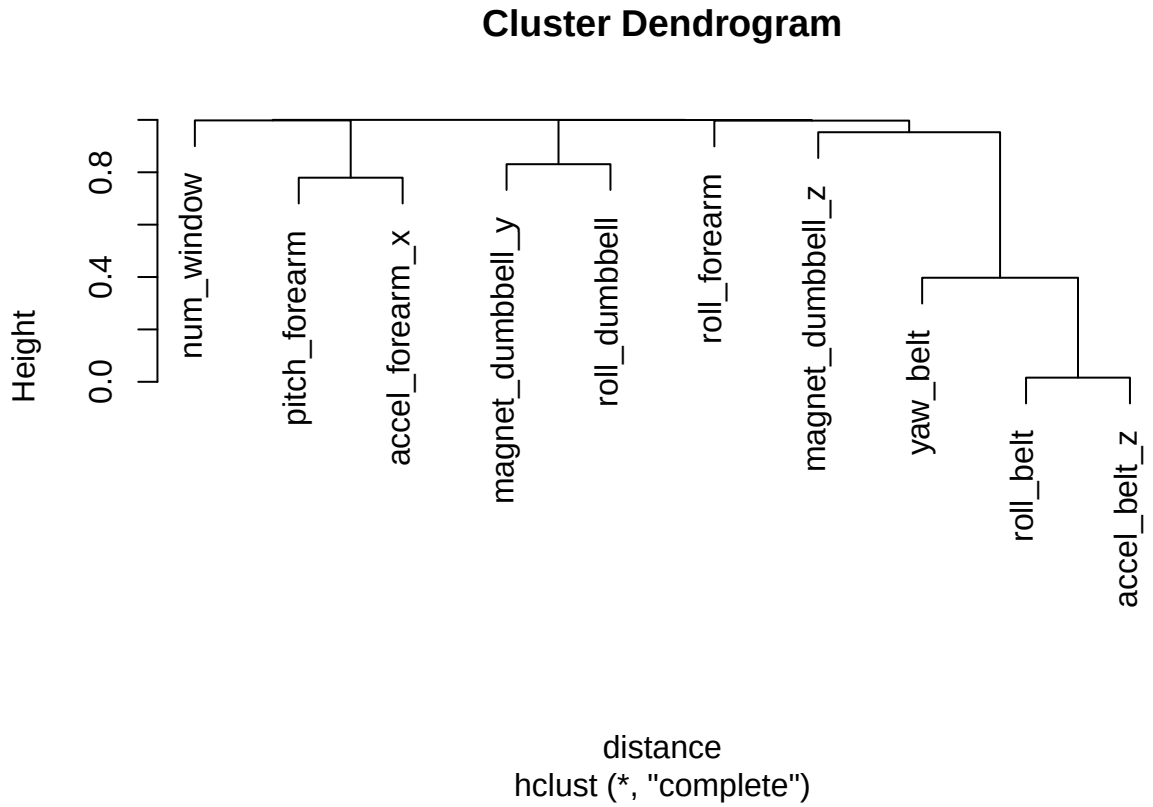
```
## rf variable importance
##
##   only 20 most important variables shown (out of 53)
##
##                      Overall
## roll_belt            100.000
## num_window            61.459
## pitch_forearm         56.559
## magnet_dumbbell_z     44.883
## magnet_dumbbell_y     40.579
## roll_forearm          37.147
## yaw_belt              26.079
## roll_dumbbell         23.102
## accel_forearm_x       19.788
## accel_belt_z          18.283
## magnet_dumbbell_x     16.672
## pitch_belt            15.964
## accel_dumbbell_y      15.479
## gyros_dumbbell_y      13.159
## magnet_forearm_z      10.617
## magnet_belt_z         10.425
## accel_dumbbell_z       9.910
## total_accel_dumbbell   9.854
## magnet_belt_y          9.848
## pitch_dumbbell         9.472
```

The top ten variables were kept for further analysis:

```
top <- cbind(name = rownames(varImp(rf_model)[[1]]), value = varImp(rf_model)[[1]])
selection <- as.character(top[order(top[,2], decreasing = T),][1:10,1])
#add classe:
selection <- c(selection, "classe")
#select only the top10 columns:
training1 <- training1[selection]
validation <- validation[selection]
```

This resulted in data sets with 10 predictor columns (plus the class column). The correlation between the remaining variables was checked once more:

```
distance <- as.dist(1-(cor(training1[,-11])^2))
plot(hclust(distance))
```



A strong correlation was found between *roll_belt* and *accel_belt_z*. Thus, *accel_belt_z* (which had a lower relative importance) was removed from the data set, leaving 9 predictors.

```
training1 <- training1[, - grep("accel_belt_z", names(training1))]
validation <- validation[, - grep("accel_belt_z", names(validation))]
```

Random Forest Model

A model was calculated from the training set with the random forest algorithm:

```
modelRF<-train(classe~., data=training1, method="rf")
```

The calculation of the model took about 7 minutes (Intel Xeon W3530 quat core CPU with 2.8 GHz, Windows 7 64-bit).

Model validation

The model was validated with the validation data set:

```
predicted <- predict(modelRF, newdata=validation)
confusionMatrix(predicted, validation$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    3    0    0    0
##          B    0 1512    0    0    2
##          C    0    3 1367    2    1
##          D    0    0    1 1284    8
##          E    0    0    0    0 1431
##
## Overall Statistics
##
##                Accuracy : 0.9975
##                  95% CI : (0.9961, 0.9984)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9968
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9960   0.9993   0.9984   0.9924
## Specificity            0.9995   0.9997   0.9991   0.9986   1.0000
## Pos Pred Value         0.9987   0.9987   0.9956   0.9930   1.0000
## Neg Pred Value         1.0000   0.9991   0.9998   0.9997   0.9983
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1927   0.1742   0.1637   0.1824
## Detection Prevalence   0.2849   0.1930   0.1750   0.1648   0.1824
## Balanced Accuracy      0.9997   0.9979   0.9992   0.9985   0.9962
```

The accuracy was 99.8%. This value was very high for machine learning predictions.

Another way to describe the predictive power of a model is to state the out of sample error rate, which is the proportion of wrongly classified cases in the validation set. To estimate this rate, the number of incorrectly predicted cases was divided by the total number of cases:

```
length(which(predicted!=validation$classe))
```

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

```
length(validation$classe)
```

```
## [1] 7846
```

```
length(which(predicted!=validation$classe)) / length(validation$classe)
```

```
## [1] 0.00254907
```

This resulted in an out of sample error rate of 0.25%. The model predicted only 20 out of 7846 cases incorrectly.

Conclusion

The initial data set was cleaned from columns consisting mostly of missing values. This resulted in a set of 54 variables, which were tested for relative importance with an initial random forest model. The most important (and not highly correlated) 9 variables were kept for the final model calculation.

The final model was calculated by the random forest method of the caret package. The model predicted the validation data set with 99.8% accuracy and an error rate of 0.25%. The model was extremely precise in the prediction, which indicated that the five classes of movement were very distinctive. To verify the model further, it would be of high interest to record new data with different participants and test the performance of the model on the new data.

Predicting the test cases

The 20 cases from the *testing* data set were predicted and displayed as a data frame.

```
data.frame(testing$problem_id, predicted = predict(modelRF, newdata=testing))
```

```
##    testing.problem_id predicted
## 1                   1         B
## 2                   2         A
## 3                   3         B
## 4                   4         A
## 5                   5         A
## 6                   6         E
## 7                   7         D
## 8                   8         B
## 9                   9         A
## 10                 10         A
## 11                 11         B
## 12                 12         C
## 13                 13         B
## 14                 14         A
## 15                 15         E
```

```
## 16                   16          E
## 17                   17          A
## 18                   18          B
## 19                   19          B
## 20                   20          B
```

The results were used to solve the Course Project Prediction Quiz of the Coursera Practical Machine learning course.