# Prediction of good method of working out!

*Fabricio Dujardin*

*10/17/2017*

## Background of the data

Six young health 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 data for this project come from this source: http://groupware.les.inf.puc-rio.br/har. Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. Participants were supervised by an experienced weight lifter to make sure the execution complied to the manner they were supposed to simulate. The exercises were performed by six male participants aged between 20-28 years, with little weight lifting experience. We made sure that all participants could easily simulate the mistakes in a safe and controlled manner by using a relatively light dumbbell (1.25kg).

Read more: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har#ixzz4p9pxiYPa

Read more: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har#ixzz4p9pXE5b7

## Loading the data

The required packages are mainly below but others are shown in the code afterwards when required. There data has been divided into 70% training and 30% testing for cross validation. The validation data is for prediction at the end of the paper.

```
library(caret)
library(ggplot2)
library(data.table)
library(cowplot)
library(knitr)
library(rpart)
library(rpart.plot)
library(pander)
library(kableExtra)
options(knitr.tableformat="html")
Testing_forprediction <- data.table(read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!","")))
Data <- data.table(read.csv("pml-training.csv",na.strings=c("NA","#DIV/0!","")))
set.seed(123)
#Partitioning the training set into 2
inTrain <- createDataPartition(Data$classe,p=0.70,list=FALSE)
Training <- Data[inTrain,]
Testing <- Data[-inTrain,]
```

## Exploring and cleaning the data

We will focus on the training data. The following steps are used to clean and adjust the data.

- Drop out variables that are near zero variability
- Drop out variables that are nearly collinear
- Drop out variables that should not be predictors out of logic
- Drop out variables that have too many NAs & have complese cases of the predictors

The aim is to have predictors that matter for the final prediction. Afterwards we need to check if the validation data has the proper classes and names as the training data.

### Near Zero Variability

Many variables have been cut off as their variability does not meet the standards required.

```
cols <- nearZeroVar(Training)
nearZeroVar(Training, saveMetrics=T)
```

```
##                         freqRatio percentUnique zeroVar   nzv
## X                        1.000000  100.00000000   FALSE FALSE
## user_name                1.110114    0.04367766   FALSE FALSE
## raw_timestamp_part_1     1.000000    6.09303341   FALSE FALSE
## raw_timestamp_part_2     1.000000   89.48096382   FALSE FALSE
## cvtd_timestamp           1.014409    0.14559220   FALSE FALSE
## new_window              46.532872    0.01455922   FALSE  TRUE
## num_window               1.000000    6.24590522   FALSE FALSE
## roll_belt                1.034375    8.08764650   FALSE FALSE
## pitch_belt               1.204918   12.17150761   FALSE FALSE
## yaw_belt                 1.077348   13.15425493   FALSE FALSE
## total_accel_belt         1.077358    0.21110868   FALSE FALSE
## kurtosis_roll_belt       2.000000    2.03829075   FALSE FALSE
## kurtosis_picth_belt      1.000000    1.70342870   FALSE FALSE
## kurtosis_yaw_belt        0.000000    0.00000000    TRUE  TRUE
## skewness_roll_belt       1.000000    2.03829075   FALSE FALSE
## skewness_roll_belt.1     1.000000    1.79806362   FALSE FALSE
## skewness_yaw_belt        0.000000    0.00000000    TRUE  TRUE
## max_roll_belt            1.142857    1.15745796   FALSE FALSE
## max_picth_belt           1.625000    0.15287181   FALSE FALSE
## max_yaw_belt             1.181818    0.39309893   FALSE FALSE
## min_roll_belt            1.125000    1.12105991   FALSE FALSE
## min_pitch_belt           2.157895    0.10919415   FALSE FALSE
## min_yaw_belt             1.181818    0.39309893   FALSE FALSE
## amplitude_roll_belt      1.304348    0.85899396   FALSE FALSE
## amplitude_pitch_belt     3.276596    0.09463493   FALSE FALSE
## amplitude_yaw_belt       0.000000    0.00727961    TRUE  TRUE
## var_total_accel_belt     1.482143    0.40765815   FALSE FALSE
## avg_roll_belt            1.090909    1.13561913   FALSE FALSE
## stddev_roll_belt         1.000000    0.45861542   FALSE FALSE
## var_roll_belt            1.648148    0.56780957   FALSE FALSE
## avg_pitch_belt           1.000000    1.26665211   FALSE FALSE
## stddev_pitch_belt        1.086957    0.28390478   FALSE FALSE
## var_pitch_belt           1.159420    0.40765815   FALSE FALSE
## avg_yaw_belt             1.125000    1.37584625   FALSE FALSE
## stddev_yaw_belt          1.875000    0.37126010   FALSE FALSE
## var_yaw_belt             1.354839    0.87355318   FALSE FALSE
## gyros_belt_x             1.072034    0.92451045   FALSE FALSE
## gyros_belt_y             1.153819    0.46589503   FALSE FALSE
```

```
## gyros_belt_z              1.064205    1.20113562   FALSE FALSE
## accel_belt_x              1.068519    1.17201718   FALSE FALSE
## accel_belt_y              1.122109    0.99730654   FALSE FALSE
## accel_belt_z              1.055921    2.11836646   FALSE FALSE
## magnet_belt_x             1.108871    2.21300138   FALSE FALSE
## magnet_belt_y             1.103604    2.08196841   FALSE FALSE
## magnet_belt_z             1.002924    3.15207105   FALSE FALSE
## roll_arm                 47.560000   17.36186940   FALSE FALSE
## pitch_arm                76.741935   20.28827255   FALSE FALSE
## yaw_arm                  30.487179   19.10897576   FALSE FALSE
## total_accel_arm           1.079872    0.48045425   FALSE FALSE
## var_accel_arm             7.000000    2.06012958   FALSE FALSE
## avg_roll_arm             49.000000    1.75438596   FALSE  TRUE
## stddev_roll_arm          49.000000    1.75438596   FALSE  TRUE
## var_roll_arm             49.000000    1.75438596   FALSE  TRUE
## avg_pitch_arm            49.000000    1.75438596   FALSE  TRUE
## stddev_pitch_arm         49.000000    1.75438596   FALSE  TRUE
## var_pitch_arm            49.000000    1.75438596   FALSE  TRUE
## avg_yaw_arm              49.000000    1.75438596   FALSE  TRUE
## stddev_yaw_arm           51.000000    1.73982675   FALSE  TRUE
## var_yaw_arm              51.000000    1.73982675   FALSE  TRUE
## gyros_arm_x               1.010753    4.61527262   FALSE FALSE
## gyros_arm_y               1.490411    2.66433719   FALSE FALSE
## gyros_arm_z               1.056848    1.70342870   FALSE FALSE
## accel_arm_x               1.133929    5.57618112   FALSE FALSE
## accel_arm_y               1.165605    3.79995632   FALSE FALSE
## accel_arm_z               1.068182    5.60529956   FALSE FALSE
## magnet_arm_x              1.101695    9.65276261   FALSE FALSE
## magnet_arm_y              1.016129    6.22406639   FALSE FALSE
## magnet_arm_z              1.012987    9.11407149   FALSE FALSE
## kurtosis_roll_arm         1.000000    1.73982675   FALSE FALSE
## kurtosis_picth_arm        1.000000    1.73254714   FALSE FALSE
## kurtosis_yaw_arm          2.000000    2.04557036   FALSE FALSE
## skewness_roll_arm         1.000000    1.74710636   FALSE FALSE
## skewness_pitch_arm        1.000000    1.73254714   FALSE FALSE
## skewness_yaw_arm          1.000000    2.05284997   FALSE FALSE
## max_roll_arm             16.333333    1.60151416   FALSE FALSE
## max_picth_arm             8.166667    1.50687923   FALSE FALSE
## max_yaw_arm               1.058824    0.36398049   FALSE FALSE
## min_roll_arm             16.333333    1.55783650   FALSE FALSE
## min_pitch_arm            12.250000    1.55783650   FALSE FALSE
## min_yaw_arm               1.052632    0.26206595   FALSE FALSE
## amplitude_roll_arm       24.500000    1.66703065   FALSE  TRUE
## amplitude_pitch_arm      17.000000    1.63791221   FALSE FALSE
## amplitude_yaw_arm         1.187500    0.36398049   FALSE FALSE
## roll_dumbbell             1.009804   86.31433355   FALSE FALSE
## pitch_dumbbell            2.000000   84.32700007   FALSE FALSE
## yaw_dumbbell              1.133333   85.69556672   FALSE FALSE
## kurtosis_roll_dumbbell    1.000000    2.05284997   FALSE FALSE
## kurtosis_picth_dumbbell   1.000000    2.06740919   FALSE FALSE
## kurtosis_yaw_dumbbell     0.000000    0.00000000    TRUE  TRUE
## skewness_roll_dumbbell    2.000000    2.07468880   FALSE FALSE
## skewness_pitch_dumbbell   1.000000    2.06740919   FALSE FALSE
## skewness_yaw_dumbbell     0.000000    0.00000000    TRUE  TRUE
```

```
## max_roll_dumbbell         1.333333    1.78350440   FALSE FALSE
## max_picth_dumbbell        1.000000    1.82718206   FALSE FALSE
## max_yaw_dumbbell          1.000000    0.45133581   FALSE FALSE
## min_roll_dumbbell         1.000000    1.79806362   FALSE FALSE
## min_pitch_dumbbell        1.000000    1.91453738   FALSE FALSE
## min_yaw_dumbbell          1.000000    0.45133581   FALSE FALSE
## amplitude_roll_dumbbell   7.000000    1.99461309   FALSE FALSE
## amplitude_pitch_dumbbell  7.000000    1.96549465   FALSE FALSE
## amplitude_yaw_dumbbell    0.000000    0.00727961    TRUE  TRUE
## total_accel_dumbbell      1.074816    0.30574361   FALSE FALSE
## var_accel_dumbbell        5.000000    1.97277426   FALSE FALSE
## avg_roll_dumbbell         1.000000    2.05284997   FALSE FALSE
## stddev_roll_dumbbell     14.000000    2.00917231   FALSE FALSE
## var_roll_dumbbell        14.000000    2.00917231   FALSE FALSE
## avg_pitch_dumbbell        1.000000    2.05284997   FALSE FALSE
## stddev_pitch_dumbbell    14.000000    2.00917231   FALSE FALSE
## var_pitch_dumbbell       14.000000    2.00917231   FALSE FALSE
## avg_yaw_dumbbell          1.000000    2.05284997   FALSE FALSE
## stddev_yaw_dumbbell      14.000000    2.00917231   FALSE FALSE
## var_yaw_dumbbell         14.000000    2.00917231   FALSE FALSE
## gyros_dumbbell_x          1.006961    1.71070831   FALSE FALSE
## gyros_dumbbell_y          1.286064    1.92181699   FALSE FALSE
## gyros_dumbbell_z          1.058962    1.43408313   FALSE FALSE
## accel_dumbbell_x          1.031111    2.95552158   FALSE FALSE
## accel_dumbbell_y          1.052023    3.27582442   FALSE FALSE
## accel_dumbbell_z          1.159509    2.89728471   FALSE FALSE
## magnet_dumbbell_x         1.068376    7.78918250   FALSE FALSE
## magnet_dumbbell_y         1.257812    6.02023732   FALSE FALSE
## magnet_dumbbell_z         1.021429    4.79726287   FALSE FALSE
## roll_forearm             11.118367   13.62742957   FALSE FALSE
## pitch_forearm            61.886364   18.99978161   FALSE FALSE
## yaw_forearm              16.106509   12.90674820   FALSE FALSE
## kurtosis_roll_forearm     1.000000    1.62335299   FALSE FALSE
## kurtosis_picth_forearm    1.000000    1.61607338   FALSE FALSE
## kurtosis_yaw_forearm      0.000000    0.00000000    TRUE  TRUE
## skewness_roll_forearm     1.000000    1.63063260   FALSE FALSE
## skewness_pitch_forearm    2.000000    1.60879377   FALSE FALSE
## skewness_yaw_forearm      0.000000    0.00000000    TRUE  TRUE
## max_roll_forearm         21.666667    1.40496469   FALSE  TRUE
## max_picth_forearm         2.708333    0.88083279   FALSE FALSE
## max_yaw_forearm           1.277778    0.26934556   FALSE FALSE
## min_roll_forearm         21.666667    1.46320157   FALSE  TRUE
## min_pitch_forearm         4.062500    0.98274732   FALSE FALSE
## min_yaw_forearm           1.277778    0.26934556   FALSE FALSE
## amplitude_roll_forearm   21.666667    1.53599767   FALSE  TRUE
## amplitude_pitch_forearm   4.785714    1.01914537   FALSE FALSE
## amplitude_yaw_forearm     0.000000    0.00727961    TRUE  TRUE
## total_accel_forearm       1.088664    0.50229308   FALSE FALSE
## var_accel_forearm         3.000000    2.08924802   FALSE FALSE
## avg_roll_forearm         32.500000    1.63063260   FALSE  TRUE
## stddev_roll_forearm      68.000000    1.61607338   FALSE  TRUE
## var_roll_forearm         68.000000    1.61607338   FALSE  TRUE
## avg_pitch_forearm        65.000000    1.63791221   FALSE  TRUE
## stddev_pitch_forearm     65.000000    1.63791221   FALSE  TRUE
```

```
## var_pitch_forearm        65.000000    1.63791221   FALSE  TRUE
## avg_yaw_forearm          65.000000    1.63791221   FALSE  TRUE
## stddev_yaw_forearm       67.000000    1.62335299   FALSE  TRUE
## var_yaw_forearm          67.000000    1.62335299   FALSE  TRUE
## gyros_forearm_x           1.074792    2.06012958   FALSE FALSE
## gyros_forearm_y           1.029412    5.24859868   FALSE FALSE
## gyros_forearm_z           1.137313    2.14020528   FALSE FALSE
## accel_forearm_x           1.047619    5.67809565   FALSE FALSE
## accel_forearm_y           1.000000    7.11217879   FALSE FALSE
## accel_forearm_z           1.185185    4.04746306   FALSE FALSE
## magnet_forearm_x          1.000000   10.61367111   FALSE FALSE
## magnet_forearm_y          1.163934   13.37264323   FALSE FALSE
## magnet_forearm_z          1.000000   11.84392517   FALSE FALSE
## classe                    1.469526    0.03639805   FALSE FALSE
```

```
length(cols)# number of variables dropped
```

```
## [1] 32
```

```
head(cols,10)
```

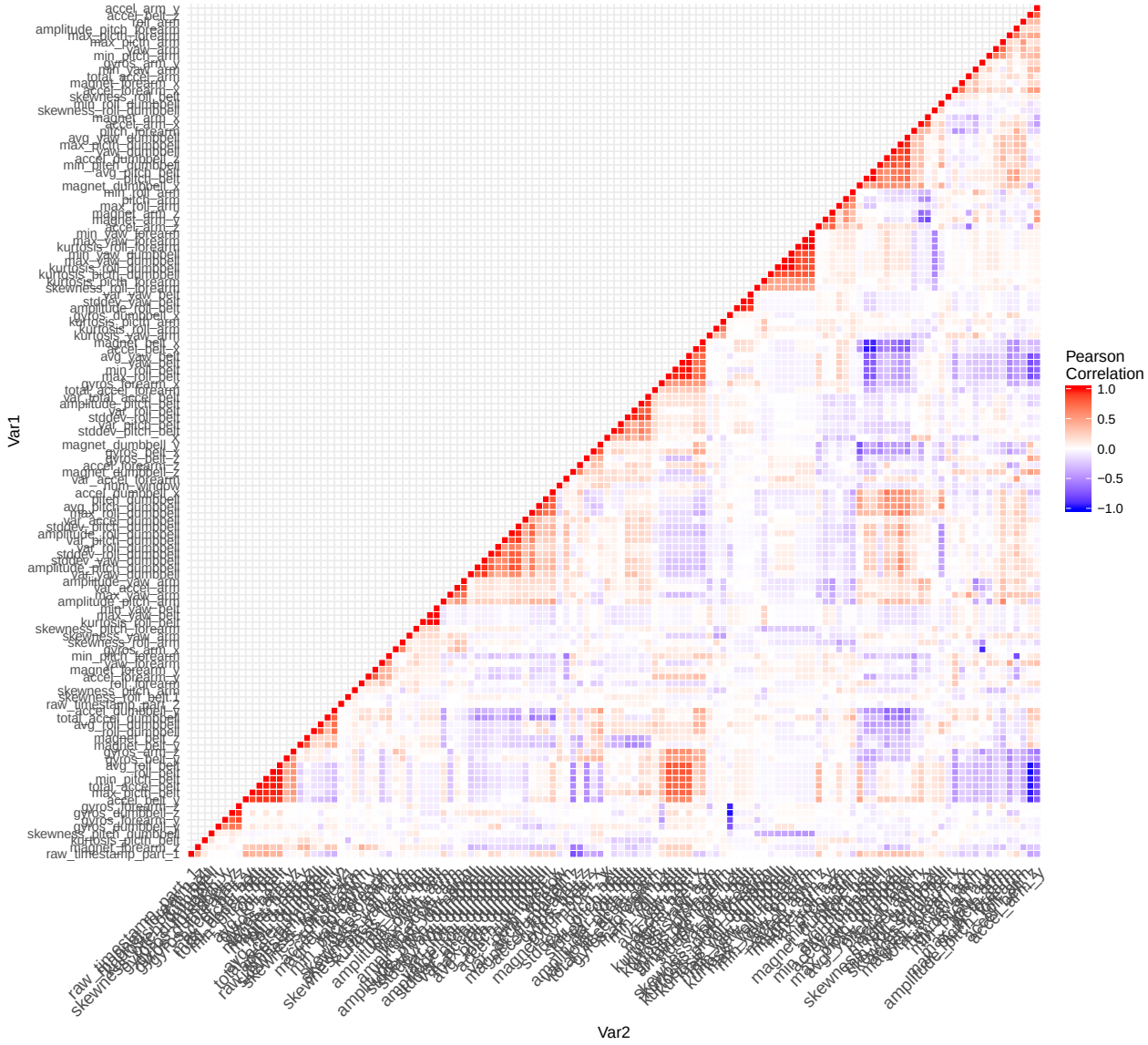
```
##  [1]  6 14 17 26 51 52 53 54 55 56
```

```
dropped1 <- Training[,.SD,.SDcols=cols]
Training1 <- Training[,.SD,.SDcols=-cols]
```

## Collinear Variables

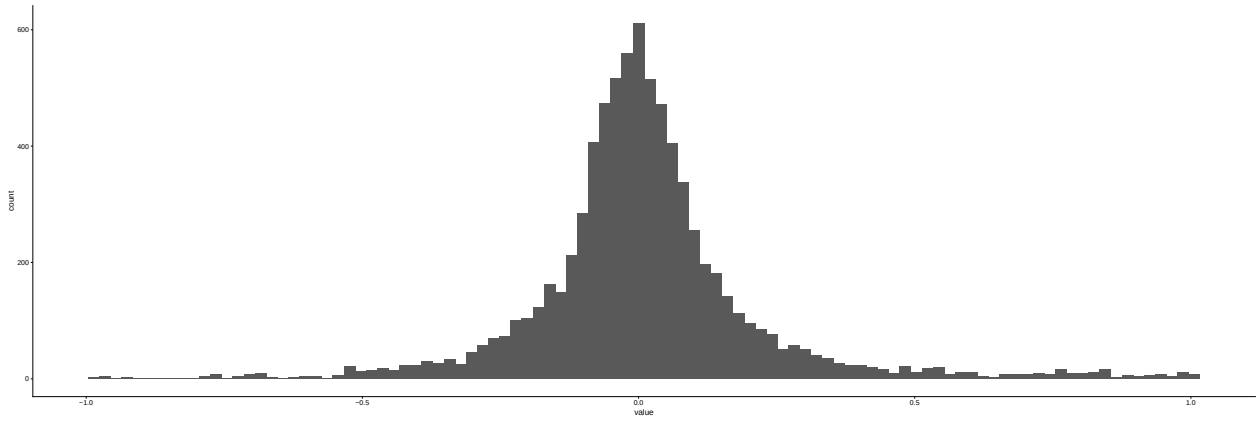
I have selected

```
#code for ggheatmap has been hidden because it takes too much space for this assignment. Kindly check g
# Print the heatmap
ggheatmap
```




```
melted_cormat <- data.table(melted_cormat)

ggplot(data=melted_cormat[value!=1,],aes(value)) + geom_histogram(bins=100)
```



```
above_0.9 <- melted_cormat[value!=1,][value>0.9|value< -0.9,]
above_0.9 <- above_0.9[order(-abs(value),Var1), ]
kable(head(above_0.9,5))
```

| Var1 | Var2 | value |
|---|---|---|
| kurtosis_roll_dumbbell | max_yaw_dumbbell | 0.9999717 |
| kurtosis_roll_dumbbell | min_yaw_dumbbell | 0.9999717 |
| kurtosis_roll_forearm | max_yaw_forearm | 0.9999501 |
| kurtosis_roll_forearm | min_yaw_forearm | 0.9999501 |
| kurtosis_roll_belt | max_yaw_belt | 0.9999291 |

I decided to drop the following variables from the Training set due to the high correlations above 0.9 and below -0.9.

```
name_todrop2<- above_0.9[,unique(Var1)]
Training2 <- Training1[,.SD,.SDcols=-as.character(name_todrop2)]
```

We have left the following amount of observations and variables.

```
dim(Training2)
```

```
## [1] 13737    98
```

## Interprettable predictors

The response is the variable class with 5 levels: A, B, C, D, E as mentioned in the background section of this assignment. The

```
Training2[,levels(classe)]
```

```
## [1] "A" "B" "C" "D" "E"
```

```
descr <- Training[,.N,by=c("classe","user_name")]
kable(dcast(descr,user_name ~ classe), caption="amount of observations per user by class")
```

Table 2: amount of observations per user by class

| user_name | A | B | C | D | |
|---|---|---|---|---|---|
| adelmo | 808 | 544 | 514 | 364 | |
| carlitos | 576 | 485 | 349 | 339 | |
| charles | 616 | 513 | 377 | 459 | |
| eurico | 619 | 426 | 352 | 404 | |
| jeremy | 836 | 331 | 456 | 369 | |
| pedro | 451 | 359 | 348 | 317 | |
| All predicto | rs see | m to h | ave an | effec | t on the classe however the time stamp is repeated in different format and t |

```
Training2 <- Training2[,.SD,.SDcols= -c("raw_timestamp_part_1","raw_timestamp_part_2")]
```

## Missing values

The missing values per variable are as follows:

```
missing <-Training2[,lapply(.SD,is.na)][,lapply(.SD,sum)][,lapply(.SD,function(x){if(x>0) x})]

length(missing)
```

```
## [1] 50
```

```
missing[,1]/dim(Training2)[1]
```

```
##    kurtosis_picth_belt
## 1:           0.9803451
```

Most of the variables have the same amount of NAs which is about 98% of the data. I further looked into it because I found it weird that the near zero variance did not discard this variables. Therefore I supposed these variables are relevant to a particular classe only. Apparently all classes have the same number of missing values per classe for hte 24 variables that have 98% of missing values. I therefore drop these variables.

```
missing
```

```
##    kurtosis_picth_belt skewness_roll_belt skewness_roll_belt.1
## 1:               13467              13455                13467
##    max_yaw_belt min_yaw_belt var_total_accel_belt var_roll_belt
## 1:        13456        13456                13448         13448
##    avg_pitch_belt var_pitch_belt avg_yaw_belt var_yaw_belt var_accel_arm
## 1:          13448          13448        13448        13448         13448
##    kurtosis_roll_arm kurtosis_picth_arm kurtosis_yaw_arm skewness_roll_arm
## 1:             13498              13499            13455             13497
##    skewness_pitch_arm skewness_yaw_arm max_roll_arm max_picth_arm
## 1:              13499            13455        13448         13448
##    max_yaw_arm min_roll_arm min_pitch_arm min_yaw_arm amplitude_pitch_arm
## 1:       13448        13448         13448       13448                13448
##    amplitude_yaw_arm kurtosis_picth_dumbbell skewness_roll_dumbbell
## 1:             13448                   13450                  13451
##    skewness_pitch_dumbbell max_roll_dumbbell max_yaw_dumbbell
## 1:                   13449             13448            13452
##    min_roll_dumbbell min_pitch_dumbbell min_yaw_dumbbell
## 1:             13448              13448            13452
##    var_accel_dumbbell avg_roll_dumbbell var_roll_dumbbell
## 1:              13448             13448             13448
##    avg_pitch_dumbbell stddev_pitch_dumbbell avg_yaw_dumbbell
## 1:              13448                 13448            13448
##    stddev_yaw_dumbbell kurtosis_picth_forearm skewness_roll_forearm
## 1:               13448                  13515                 13513
##    skewness_pitch_forearm max_picth_forearm max_yaw_forearm
## 1:                  13515             13448           13514
##    min_pitch_forearm min_yaw_forearm amplitude_pitch_forearm
## 1:             13448           13514                  13448
##    var_accel_forearm
## 1:             13448
```

```
M<-Training2[,.SD,.SDcols=c("classe",names(missing))]
kable(M[,lapply(.SD,is.na),by=classe][,lapply(.SD,sum,na.rm=T),by=classe][,1:2])
```

| classe | kurtosis_picth_belt |
|---|---|
| A | 3838 |
| B | 2604 |
| C | 2348 |

| classe | kurtosis_picth_belt |
|---|---|
| D | 2209 |
| E | 2468 |

### Selected variables to use as predictors.

We are left with fewer variables in the training data.

```
Training3 <- Training2[,.SD,.SDcols= -names(missing)]
dim(Training3)
```

```
## [1] 13737    46
```

```
dim(Training3) == dim(na.omit(Training3))
```

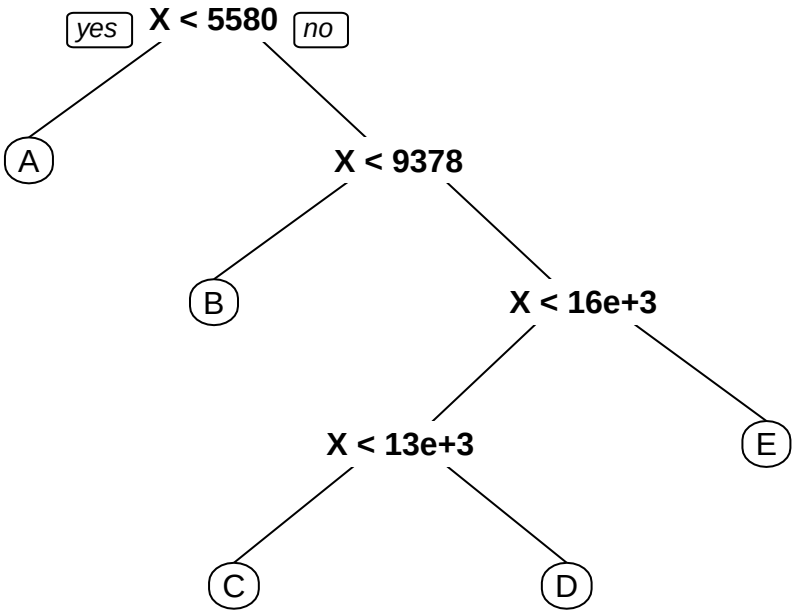
```
## [1] TRUE TRUE
```

## Models used with adjusted Training data

- Decision tree

We fit a predictive model for activity recognition using the decision tree algorithm.

```
tree_ <- rpart(classe ~ ., data=Training3, method="class")
prp(tree_)
```




The performance of the model with the validation data is as follows:

```
tree_out <- predict(tree_, Testing, type = "class")
tree_ct <- confusionMatrix(Testing$classe, tree_out)
tree_ct
```

```
## Confusion Matrix and Statistics
##
```

Table 4: confusion table of the tree model

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 1674 | 0 | 0 | 0 | 0 |
| B | 0 | 1139 | 0 | 0 | 0 |
| C | 0 | 0 | 1026 | 0 | 0 |
| D | 0 | 0 | 0 | 964 | 0 |
| E | 0 | 0 | 0 | 0 | 1082 |

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
##          B    0 1139    0    0    0
##          C    0    0 1026    0    0
##          D    0    0    0  964    0
##          E    0    0    0    0 1082
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9994, 1)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```
accuracy <- postResample(tree_out, Testing$classe)
#show table
tab <- tree_ct
kable(tab[2],caption="confusion table of the tree model")
```

```
rm(tree_out)
```

The Accuracy is:

```
kable(tab$overall)
```

| | |
|---|---|
| Accuracy | 1.0000000 |
| Kappa | 1.0000000 |
| AccuracyLower | 0.9993734 |
| AccuracyUpper | 1.0000000 |

Table 6: confusion table of the Random Forest model

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 1674 | 0 | 0 | 0 | 0 |
| B | 0 | 1139 | 1 | 0 | 0 |
| C | 0 | 0 | 1025 | 0 | 0 |
| D | 0 | 0 | 0 | 964 | 0 |
| E | 0 | 0 | 0 | 0 | 1082 |

| | |
|---|---|
| AccuracyNull | 0.2844520 |
| AccuracyPValue | 0.0000000 |
| McnemarPValue | NaN |

- Random Forest

It is quite accurate since at each split it has bootstrap variables. It grows multiple trees and vote. We will use 4 fold cross validation when applying the algorithm.

```
#rf - random forest
rf_ <- train(classe ~ ., data=Training3, method="rf", verbose=FALSE, trControl=trainControl(method ="cv
#predict
rf.out <- predict(rf_,newdata=Testing)
#contigency table
rf.ct <- confusionMatrix(rf.out,Testing$classe)
#show table
tab <- rf.ct
kable(tab[2],caption="confusion table of the Random Forest model")
```

```
rm(rf_out)
rm(rf_ct)
```

The Accuracy is:

```
kable(tab$overall)
```

| | |
|---|---|
| Accuracy | 0.9998301 |
| Kappa | 0.9997851 |
| AccuracyLower | 0.9990536 |
| AccuracyUpper | 0.9999957 |
| AccuracyNull | 0.2844520 |
| AccuracyPValue | 0.0000000 |
| McnemarPValue | NaN |

# Prediction/ Forecasting of the 20 classes

We use the validation data from the first section and predict the 20 classes with all the models. As they were all equally good. But before we make sure that the predicted data has the same format as the training and testing. We therefore need to coerce it into the same type of data.

```
Model.tree <- predict(tree_,Testing_forprediction)
Model.rf <- predict(rf_,Testing_forprediction)

result_predictions = data.frame(Model.tree)
kable(data.frame(result_predictions), caption=" results for the final quiz")
```



Table 8: results for the final quiz

| A | B | C | D | E |
|---|---|---|---|---|

| A | B | C | D | E |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |

```
result_predictions = data.frame(Model.rf)
kable(data.frame(result_predictions), caption=" results for the final quiz")
```

Table 9: results for the final quiz

| Model.rf |
|---|
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |
| A |

```
write.table(result_predictions, "results.csv")
```