Predicting how well 'weight lifting exercises' are performed

Executive summary

In this study, we build a model so as to predict how well 'weight lifting exercises' are performed by wearer. Given the foreseen non linearity and given the high number of features, we choose to estimate a Random Forest model.
The test error rate is very low (0.23%).
We also show the list of the most important variables in the model.

First settings

Here we set up the default options and load packages.

```
knitr::opts_chunk$set(echo=TRUE, warning=FALSE, message=FALSE)
library(caret); library (randomForest)
```

Then, we load both training and testing data.

```
training <- read.csv("pml-training.csv")
testing <- read.csv("pml-testing.csv")
```

We finally delete unuseful variables, i.e. those for which we have too many missing values. We also delete unreleavant features like

```
training <- training[ ,-grep("^(kurtosis|skewness|max|amplitude|var|avg|stddev|min)", names(training))]
training <- training[ ,-grep("(X|user_name|timestamp)", names(training))]

testing  <- testing[ ,-grep("^(kurtosis|skewness|max|amplitude|var|avg|stddev|min)", names(testing))]
testing <- testing[ ,-grep("(X|user_name|timestamp)", names(testing))]
```

Model decription and model choice

We build a random forest model. It allows to fit different types of relationship between Y and X, especially when the model is supposed to be non linear. It also performs well when there are many variables. By nature, the model exludes unrelevant variables, so there is no risk of overfitting. Finally, the OOB error rate is a good proxy of test error. We use 50 trees, which is enough, as shown in the next outputs.

Model fitting

Given the foreseen non linearity and gievn the high number of features, we choose to estimate a Random Forest model based on 50 trees.

```
set.seed(123)
modFit <- randomForest(classe ~ ., data = training, na.action=na.omit, ntree=50, importance=TRUE)
```

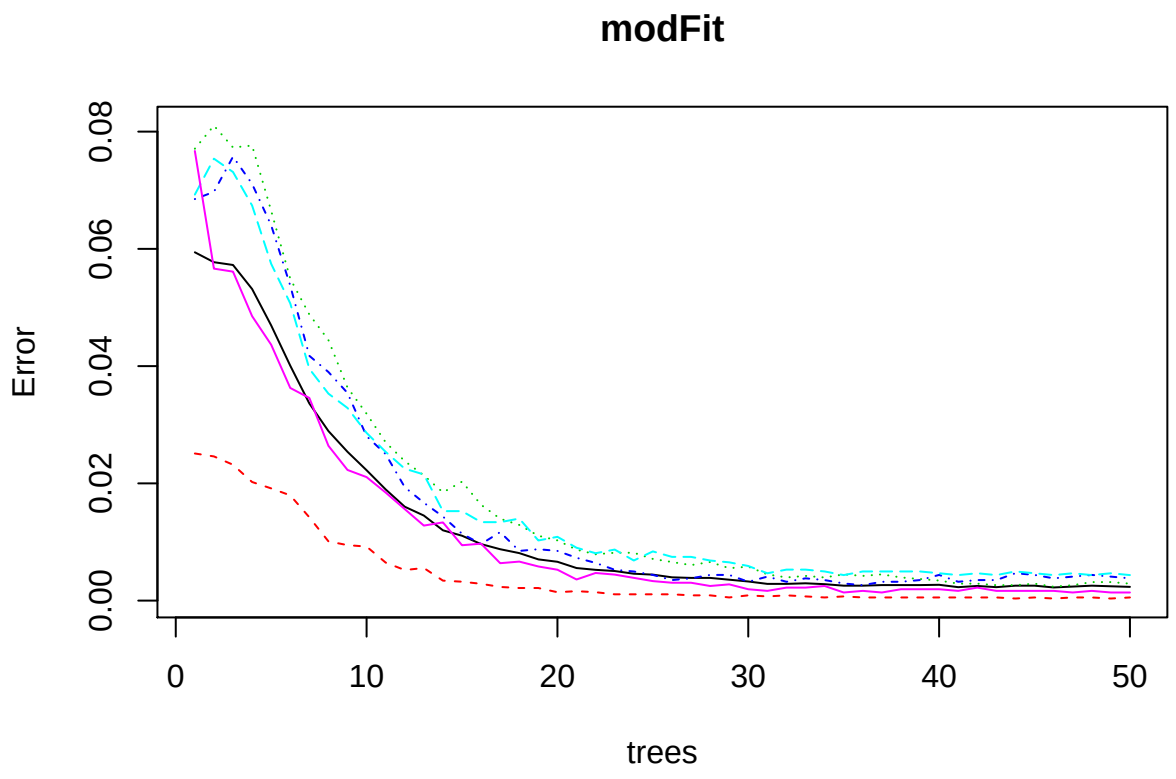
The accuracy is really high. The out-of-bound error, which is a good estimate of the test error is only 0.23%:

```
modFit
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training, ntree = 50,      importance = TRUE, na.action =
##                Type of random forest: classification
##                      Number of trees: 50
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.23%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 5577    3    0    0    0 0.0005376344
## B    7 3786    4    0    0 0.0028970240
## C    0   12 3409    1    0 0.0037989480
## D    0    0   12 3202    2 0.0043532338
## E    0    1    0    4 3602 0.0013861935
```

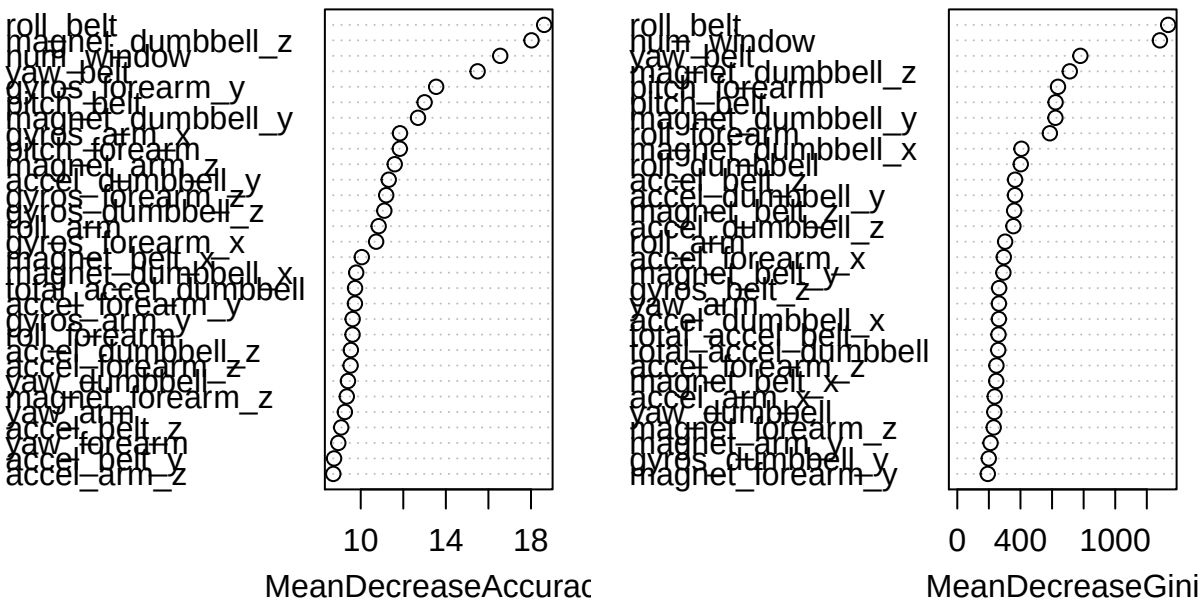
The error rate is decreasing very quickly and suggest that the marginal gain of growing more than 50 trees is small

```
plot(modFit)
```



Here we show the top features, i.e. those who reduce the most the model accuracy when removed from the model:

```
varImpPlot (modFit)
```



Cross-validaton and expected out of sample error

Cross-validation is not necessary to estimate the test error when growing a random forest (RF). Each tree is built on a reduced sample. Then, test error (called out-of-bag (OOB)) on each tree can be calculated by using the prediction on the out-of-sample data on this tree. Finally, OOB error are averaged over all trees.

Prediction on the test set

The code to be run for predicting 'class' on the test set is:
predict(modFit, newdata=testing)
Note that testing and training variable formats need to be aligned first (see magnet_dumbbell_z, magnet_forearm_y, magnet_forearm_z), as well as factor levels.

Main conclusion

We built a good model for predicting how well 'weight lifting exercises' are performed by wearer.
The model predictions are right in 99.77 cases out of 100. The model is a random forest and can be used to predict the outcome on new data sets.