Assignment 4 - Measuring movements

Goal

Goal of the project is to predict how participants performed a barbell lifts correctly, based on the activity measured by the accelorometers on the belt, forearm, arm and dumbell.

Method

For this project, a training set is used from http://groupware.les.inf.puc-rio.br/har. This dataset consists of 160 variables and 19622 variables.

Cleaning up the data is done by removing the columns that contain values which that only have one unique value or columns that have a very few unique values relative to the number of samples. Next to that, the columns which contain NA-values are removed from the dataset. As the dataset contains columns which should not be used for the model (identifier columns, names of the participants, time), these columns are removed from the dataset. After cleaning up, there are 53 columns left.

After cleaning up the data, the data is split up in a training set (3/4 of the dataset) and a test set (1/4 of the dataset) which is used for cross validation.

The models will be build on the training set and tested on the testset (the 1/4 part of the original training set). Based on the confusion matrices and the in sample error, the best model is chosen.

As I don't know whether the data is good, it is difficult to do en estimation of the in sample and the out of sample erorr. I think we might expect an in sample error lower than 0.3, otherwise there are a lot of incorrect labels created by the model. The out of sample is higher than the in sample error, I think. Otherwise, there would be overfitting.

```r
#load dataset
setwd("~/coursera")
pmltraining<-read.csv("pml-training.csv")

#create dataframe columns with nearzerovalues
nzvar<-as.data.frame(nearZeroVar(pmltraining,names=TRUE,saveMetrics = FALSE))

#rename the column
colnames(nzvar)<-c("column")

#subset data, only columns that are not in the nearzerovalues
data<-pmltraining[,-which(names(pmltraining) %in% nzvar$column)]
#rm(nzvar)

#subset data, only columns that do not contain NA-values
navar<-as.data.frame(names(which(sapply(data, anyNA))))
colnames(navar)<-c("column")

data<-data[,-which(names(data) %in% navar$column)]
#rm(navar)

#remove columns that should not be used (the identifier column, name of participants, the time)
idvar <- as.data.frame(grep("X|name|timestamp|window", colnames(data), value=T))
colnames(idvar)<-c("column")
```

```
data<-data[,-which(names(data) %in% idvar$column)]
#rm(idvar)

#split data in trainin and testset
inTrain = createDataPartition(data$classe, p = 3/4)[[1]]

training = data[ inTrain,]
testing = data[-inTrain,]
#rm(inTrain)
```

The created trainingset consists of 14718 rows. The created testset consists of 4904 rows. The datasets consists of 53 columns of which 1 is the column 'classe'.

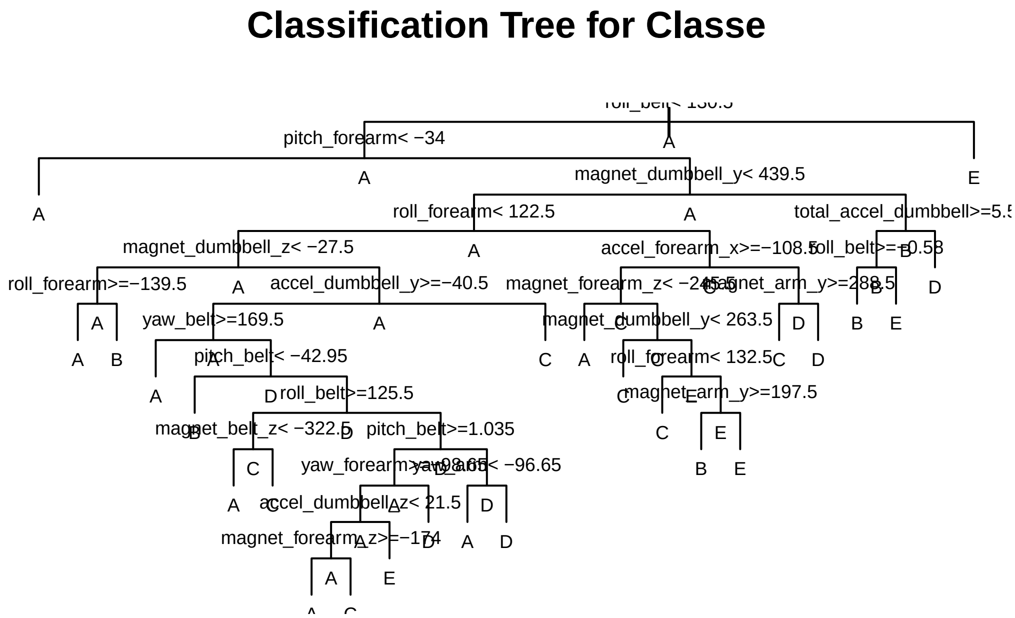
Two models are created. First, a decision tree model is trained, with the rpart package. Next to that a random forest model is trained with the randomForest package. After training the model, the model is applied to the training set, to find out how many of the results are predicted correctly by the model. The confusion matrices and accuracy of both models will be used to choose the final model.

Model

```
#create a model (decision tree)
modeldt<-rpart(classe~., data=training, method = "class")

#use the model to predict the results of the training set
preddt<-predict(modeldt, training, type="class")

#create confusion matrix and calculate the accuracy
cmdt<-table(training$classe,preddt)
accdt<-sum(diag(cmdt))/sum(cmdt)

#create a model with caret package ()
modelrf<-randomForest(classe~., data=training)
predrf<-predict(modelrf, training, type="class")

#create confusion matrix and calculate the accuracy
cmrf<-table(training$classe,predrf)
accrf<-mean(predrf == training$classe)
```

The decision tree model generates a decision tree. This model is shown below.

```
#plot the decision tree
plot <- plot(modeldt, uniform=TRUE,
    main="Classification Tree for Classe")
text(modeldt, use.n=FALSE, all=TRUE, cex=.6)
```



Accuracy and the in sample error

Below, the confusion matrices from the both models are shown. The rows show the reference values (the original values), the columns show the predicted values. The decision tree model has an accuracy of 0.7637587, the random forest model has an accuracy of 1. These accuracy measures are an indicator of the in sample error. The in sample error is the error rate you get when you apply a model on the same dataset as you based the model on. The higher the accuracy, the lower the in sample error. We define the in sample error as 1-accuracy. Therefore, the in sample error of the decision tree model is 0.2362413 and the in sample error of the random forest model is 0.

```
cmdt %>% kable(caption = "Confusion Matrix Decision Tree Model")
```

Table 1: Confusion Matrix Decision Tree Model

	A	B	C	D	E
A	3886	108	99	53	39
B	455	1548	419	173	253
C	41	205	2092	181	48
D	138	210	203	1671	190
E	29	232	205	196	2044

```
cmrf %>% kable(caption = "Confusion Matrix Random Forest Model")
```

Table 2: Confusion Matrix Random Forest Model

	A	B	C	D	E
A	4185	0	0	0	0
B	0	2848	0	0	0
C	0	0	2567	0	0
D	0	0	0	2412	0
E	0	0	0	0	2706

As the random forest model has a perfect accuracy, we choose this model.

The next step is to use the model for predicting the classe on the test set. In this way, we can find out how well the trained model performs on a new dataset. In order to to this, we apply the model trained on the training set to the test set and have a look at the confusion matrix and the accuracy.

```
#predict values
predrftest<-predict(modelrf, testing, type="class")

#show confusion matrix and calculate the accuracy
cmrftest<-table(testing$classe,predrftest)
accrftest<-mean(predrftest == testing$classe)
```

Accuracy and the out of sample error

Below, the confusion matrix from the random tree model on the testing set is shown. This model has an accuracy of 0.9961256 on the testing set. The out of sample error is the error rate you get when you apply the model to a new dataset. As this is the case in this part, we can calculate the out of sample error as 1-accuracy. The out of sample error is 0.0038744,

```
cmrftest %>% kable(caption = "Confusion matrix on test set")
```

Table 3: Confusion matrix on test set

	A	B	C	D	E
A	1393	2	0	0	0
B	0	949	0	0	0
C	0	4	851	0	0
D	0	0	9	793	2
E	0	0	0	2	899

Conclusion

The random forest model which is trained on the training set performs well on the test set. Because of that, we conclude that this model can be used for predicting how participants performed a barbell lifts.