Prediction Assignment Coursera

A Axenholm

1/24/2020

Executive summary

This report develops a model to classify wheter a excercise is done in a proper way (Class A) or is done in some of the inaccurate ways (Class B-E) based on data from individual training devices. Information about the data can be found at: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

```
library(caret);library(dplyr);library(randomForest);library(rpart);library(rattle);library(corrplot)
```

```
trainURL <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
trainFile <- basename(trainURL)
testURL <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
testFile <- basename(testURL)
setwd("/Users/ankan/documents/data")
if(!file.exists(trainFile)) download.file(trainURL, destfile = trainFile)
training<-read.csv(trainFile, sep=',', header=TRUE)  ##, na.strings=c("#DIV/0!")
if(!file.exists(testFile)) download.file(testURL, destfile = testFile)
testing<-read.csv("/Users/ankan/Documents/data/pml-testing.csv", sep=',', header=TRUE)
```

Data

From the original datasets all calculated variables and those that aren't measurment data from the accelerometers are deleted except for the classification variable "classe" in the pml-training file.

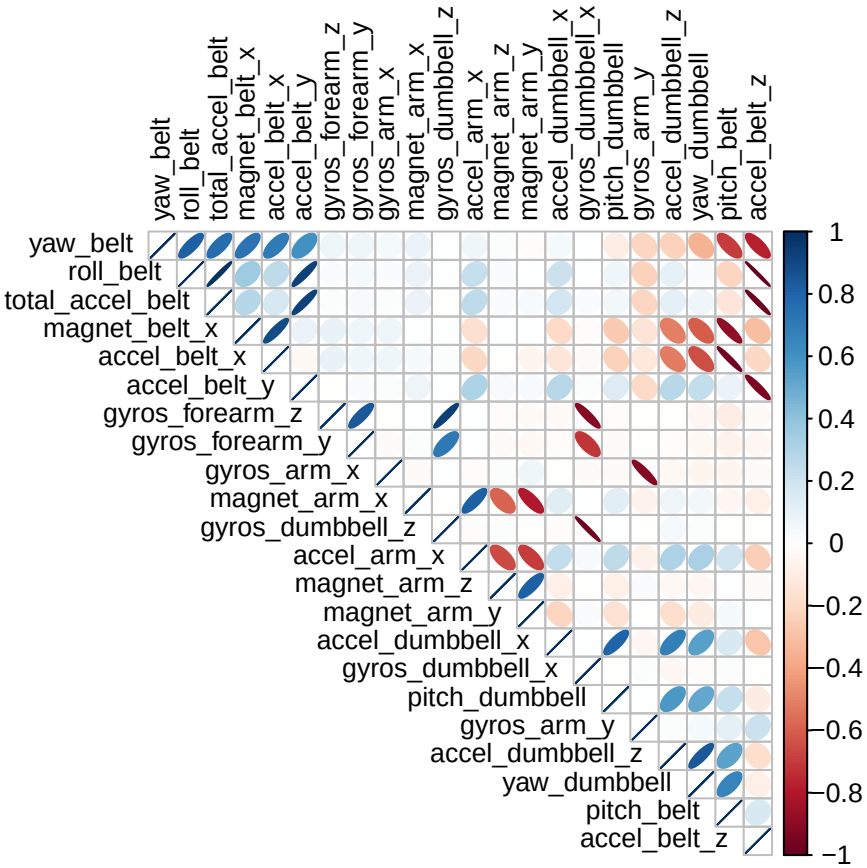
```
training <- training %>%
        select(-starts_with("kurtosis_")) %>%
        select(-starts_with("skewness_")) %>%
        select(-starts_with("min_")) %>%
        select(-starts_with("max_")) %>%
        select(-starts_with("stddev_")) %>%
        select(-starts_with("var_")) %>%
        select(-starts_with("avg_")) %>%
        select(-starts_with("amplitude_"))
training<-training[,8:ncol(training)]
testing <- testing %>%
        select(-starts_with("kurtosis_")) %>%
        select(-starts_with("skewness_")) %>%
        select(-starts_with("min_")) %>%
        select(-starts_with("max_")) %>%
        select(-starts_with("stddev_")) %>%
        select(-starts_with("var_")) %>%
        select(-starts_with("avg_")) %>%
        select(-starts_with("amplitude_"))
testing<-testing[,8:ncol(testing)]
```

By checking the near zero varians function all the remaining 52 measure variables have varians within them and are not concidered excluding by that reason.

```
nearZeroVar(training[,-53],saveMetrics=TRUE)
```

To reduce problems with overfitting I look a bit closer at the correlation between the variables. The correlation plot shows high correlated variables in dark color (blue for positive and red for negative correlation). As can be seen a lot of variables have high (over 0.8 or under-0.8) correlation and quiet a few are close to 1 (thin blue or red line).

```
## Calculate correlations absolute values
C<-abs(cor(training[,1:52]))
## Set correlation with themselves to zero
diag(C)<-0
## Create list of the variables with high correlation
N<-which(C>0.8,arr.ind=T, useNames=T)
## Select the highly correlated variables from the training set and calculate their correlations.
Ntraining <- cor(training[,unique(N[,1])])
## plot the correlations
corrplot(Ntraining, order = "FPC", method = "ellips", type = "upper", tl.cex = 0.8, tl.col = rgb(0, 0, 
```



If needed a principal component analysis of the above variables could be used if we dont find variance enough to explain "classe" without them. In a first analysis I choose to exclude them as there's still a lot of variables (31) left to make a model with.

```
training<-training[,-unique(N[,1])]
testing<-testing[,-unique(N[,1])]
```

I'll split the "pml-training" dataset and use for training and testing and then use the "pml-testing" data for validation.

```
part <- createDataPartition(y=training$classe,p=0.7, list=FALSE)
trainset <- training[part,]
testset <- training[-part,]
rbind("training set" = dim(trainset),"testing set" = dim(testset), "validation set" = dim(testing))
```

```
##                 [,1] [,2]
## training set   13737   31
## testing set     5885   31
## validation set    20   31
```

Decision tree model

First I try the rpart decision tree model which gives a sample error rate of 0.5, which is not that good.

```
set.seed(123)
DTM<-train(factor(classe)~.,data=trainset, method="rpart")
DTMpred<-predict(DTM, newdata=testset)
## Calculate Sample error
1-as.numeric(confusionMatrix(DTMpred,testset$classe)$overall[1])
```

```
## [1] 0.5174172
```

Random Forest model

For the Random Forest model I use crossvalidation with 5 folds. This gives a model with sample error rate 0.012 which hopefully is enough to solve the quiz in the validation set.

```
set.seed(1234)
trC <- trainControl(method="cv", number=5)
modFit<-train(factor(classe)~.,data=trainset, method="rf", trControl= trC, verbose=FALSE)
predictions<-predict(modFit, newdata=testset)
accmodFit<-as.numeric(confusionMatrix(predictions,testset$classe)$overall[1])
SE<-1-accmodFit
SE
```

```
## [1] 0.01240442
```

Validation

From the cleaned pml-testing set I exclude the last "problem_id" variable before making the predictions

```
predQuiz <- predict(modFit, newdata = testing[,1:30])
predQuiz
```

```
##  [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
```