Practical Machine Learning - Course Project

Stefan

24 11 2020

Installing packages

```
library(caret)
library(knitr)
library(randomForest)
library(corrplot)
library(rpart)
library(rattle)
library(rpart.plot)
library(e1071)
library(ggplot2)
library(cowplot)
library(randomForest)
```

Loading the training data

```
Train_url <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-
training.csv"
Test_url  <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-
testing.csv"

train_data <- read.csv(url(Train_url))
test_data <- read.csv(url(Test_url))
```

Splitting the training data for further analysis

```
SubGroups=createDataPartition(train_data$classe, p=0.7, list=FALSE)

Training <- train_data[SubGroups, ]
Testing <- train_data[-SubGroups, ]
```

Removing Variables with near zero Variance

Since some variables have a near zero variance, they are excluded for further analysis

```
NZV <- nearZeroVar(Training)

Trainset <- Training[ ,-NZV]
Testset <- Testing[ ,-NZV]
```

Removing variables mostly NA

Since some variables have many NAs, these variables are also excluded.

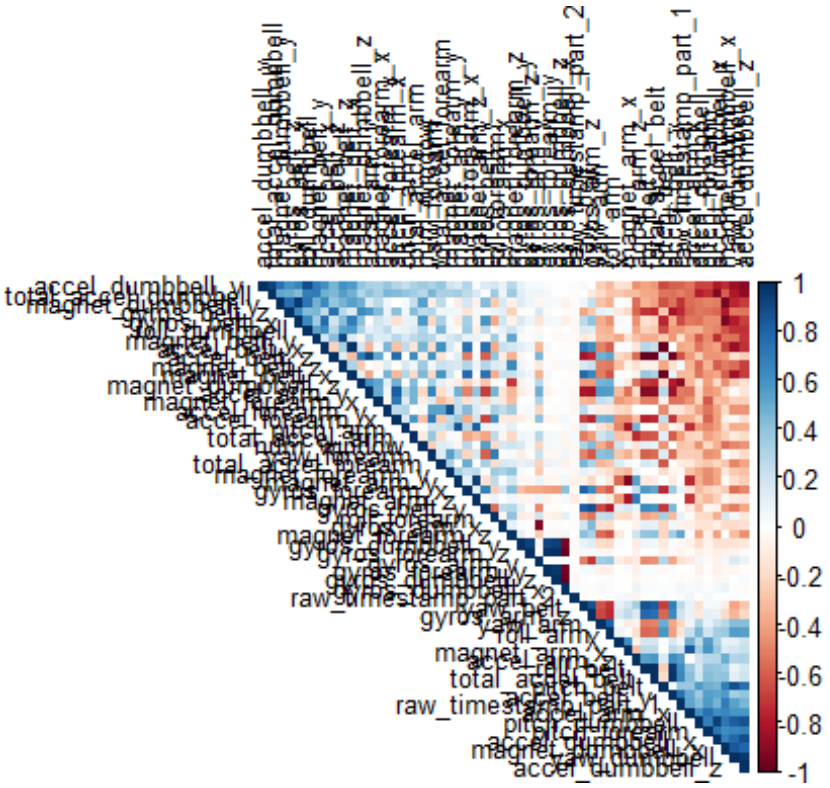
```
label <- apply(Trainset, 2, function(x) mean(is.na(x))) > 0.95
Train <- Trainset[, -which(label, label == FALSE)]
Test <- Testset[, -which(label, label == FALSE)]
```

Create correlation matrix and exclude high correlated variables

Before the analysis we check the individual variables for multicollinearity. Variables with high multicolliniarity are excluded in order not to falsify the results

```
cor.matrix_train <- cor(Train[sapply(Train, is.numeric)])

cor_mat_train <- cor(cor.matrix_train[, -53])
corrplot(cor_mat_train, order = "FPC", method = "color", type = "upper",
         tl.cex = 0.8, tl.col = rgb(0, 0, 0))
```

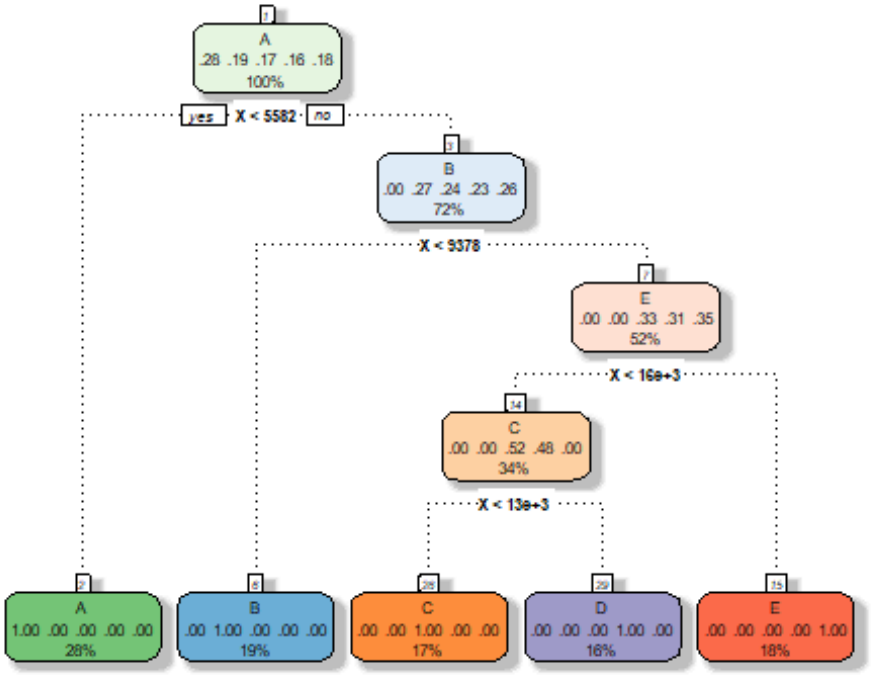



```
c <- findCorrelation(cor.matrix_train, cutoff = .90)
Trainset_prefinal <- Train[,-c]
```

Decission Tree

Beacouse we try to predict classes in form of groups and not numeric values, the first analysis is performed using a decision tree.With the help of the Confusion Matrix, the prediction accuracy should be better illustrated

```
set.seed(123)
DT_Model <- rpart(classe ~., data = Trainset_prefinal, method = "class")
fancyRpartPlot(DT_Model)
```




```
predictDT <- predict(DT_Model,Testset, type = "class")
ConMatDT <- confusionMatrix(predictDT, Testset$classe)
ConMatDT
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    1    0    0    0
##          B    0 1138    0    0    0
##          C    0    0 1025    0    0
##          D    0    0    1  963    0
##          E    0    0    0    1 1082
##
## Overall Statistics
##
##                Accuracy : 0.9995
##                  95% CI : (0.9985, 0.9999)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9994
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
```

```
## Sensitivity            1.0000   0.9991   0.9990   0.9990   1.0000
## Specificity            0.9998   1.0000   1.0000   0.9998   0.9998
## Pos Pred Value         0.9994   1.0000   1.0000   0.9990   0.9991
## Neg Pred Value         1.0000   0.9998   0.9998   0.9998   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1934   0.1742   0.1636   0.1839
## Detection Prevalence   0.2846   0.1934   0.1742   0.1638   0.1840
## Balanced Accuracy      0.9999   0.9996   0.9995   0.9994   0.9999
```

Regarding the confusion Matrix, we get really good predictions with an Accurancy of 99,95 percent and and just a single missclassification

Random Forrest

Random Forrest is chosen as the second Modell, because although it is usually more difficult to interpret, it provides better predictions. The Confusion Matrix should also clarify the accuracy here

```
set.seed(123)
RF <- randomForest(classe ~. , data= Trainset_prefinal, method="parRF")
predict_RF <- predict(RF, Testset, type = "class")

conMatRF <- confusionMatrix(predict_RF, Testset$classe)
conMatRF

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    1    0    0    0
##          B    0 1138    0    0    0
##          C    0    0 1026    0    0
##          D    0    0    0  964    0
##          E    0    0    0    0 1082
##
## Overall Statistics
##
##                Accuracy : 0.9998
##                  95% CI : (0.9991, 1)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9998
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9991   1.0000   1.0000   1.0000
## Specificity            0.9998   1.0000   1.0000   1.0000   1.0000
```

```
## Pos Pred Value         0.9994   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   0.9998   1.0000   1.0000   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1934   0.1743   0.1638   0.1839
## Detection Prevalence   0.2846   0.1934   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9999   0.9996   1.0000   1.0000   1.0000
```

Looking at the confustion matrix, we can see that there is no misscalssification in the Random Forrest Modell with an Accurancy of 99,98 percent, which is exceptionally high.

Final Prediction

Since, as expected, the random forest model provides a better prediction than the decision tree, the final test set is predicted using the random forest model

```
RF_final <- predict(RF,test_data)
RF_final
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A  A
## Levels: A B C D E
```