Prediction Assignment

Ekaterina Voronina

17 09 2020

In this assignment we have 19622 observations from weight lifting exercises. Our outcome is a factor variable called 'classe'. In this dataset 6 young healthy participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different ways which are marked as A, B, C, D and E. So we should keep in mind if the condition of experiment will change it can change the outcome

I used and compared two models to see which one has bigger accuracy percentage: decision tree model and random forest model. 70% of the total training observations were used to build the models and the rest of 30% of the observations were used for model validation. Aslo the plots built in the analysis showing top 20 variables impact on the outcome and the accuracy in predicted and observed sets.

Downloading data and preparing libraries

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.2
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.2
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following object is masked from 'package:randomForest':
##
##     margin
```

```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 4.0.2
```

```
library(rpart.plot)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v tibble  3.0.1     v dplyr   0.8.5
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
## v purrr   0.3.4
```

```
## -- Conflicts ------------------------------------------------------------------------------------ t
## x dplyr::combine()  masks randomForest::combine()
## x dplyr::filter()   masks stats::filter()
## x dplyr::lag()      masks stats::lag()
## x purrr::lift()     masks caret::lift()
## x ggplot2::margin() masks randomForest::margin()
```

```
url_train <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
download.file(url_train,'pml-training.csv')

url_test <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'
download.file(url_test,'pml-testing.csv')

train_data <- read.csv("pml-training.csv", na.strings = c("NA", "#DIV/0!", ""))
test_data  <- read.csv("pml-testing.csv", na.strings = c("NA", "#DIV/0!", ""))

dim(train_data)
```

```
## [1] 19622   160
```

```
sum(is.na(train_data))
```

```
## [1] 1925102
```

```
dim(test_data)
```

```
## [1]  20 160
```

```
sum(is.na(test_data))
```

```
## [1] 2000
```

Cleaning the data by removing NA columns and columns which are not useful for the assignment

```r
no_na <- complete.cases(t(train_data)) & complete.cases(t(test_data))
train_data <- train_data[,no_na]
test_data  <- test_data[,no_na]
sum(is.na(train_data))
```

```
## [1] 0
```

```r
sum(is.na(test_data))
```

```
## [1] 0
```

```r
train_data <- train_data[,-c(1:7)]
test_data  <- test_data[,-c(1:7)]
```

Making the analysis reproducible

```r
set.seed(123)
```

Making data slicing by spliting the training data into training(train) set and validation(test) set.

```r
samples <- createDataPartition(y=train_data$classe, p=0.7, list=FALSE)
sub_train <- train_data[samples, ]
sub_test <- train_data[-samples, ]
dim(sub_train)
```

```
## [1] 13737    53
```

```r
dim(sub_test)
```

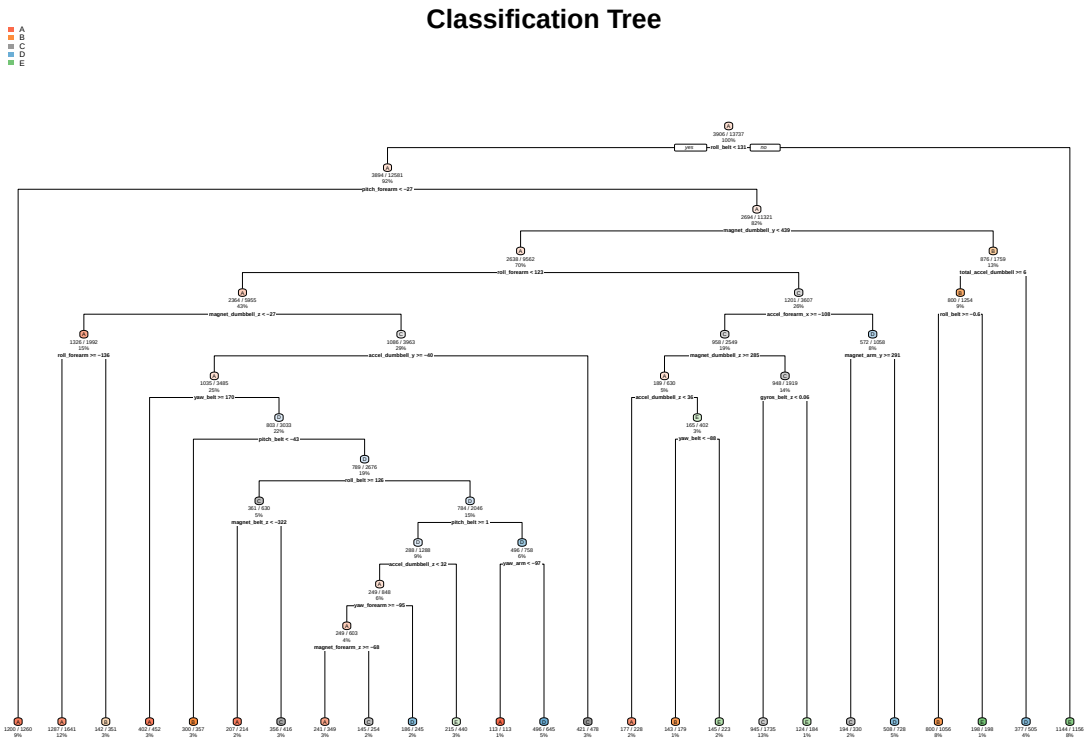
```
## [1] 5885   53
```

Building a first model usuing a decision tree

```r
sub_test$classe <- as.factor(sub_test$classe)
model1 <- rpart(classe ~ ., data = sub_train, method="class")
prediction1 <- predict(model1, sub_test, type = "class")
accuracy <- postResample(prediction1, sub_test$classe)
```

Ploting the Decision Tree

```
rpart.plot(model1, main="Classification Tree", extra=102, under=TRUE, faclen=0)
```



Testing the results using confusion matrix function:

```
confusionMatrix(sub_test$classe, prediction1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1552   48   39   24   11
##          B  174  588  220   83   74
##          C   18   43  888   75    2
##          D   60   63  100  651   90
##          E    6   64  148   86  778
##
## Overall Statistics
##
##                Accuracy : 0.7573
##                  95% CI : (0.7462, 0.7683)
##     No Information Rate : 0.3076
```

```
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6926
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8575  0.72953   0.6366   0.7084   0.8147
## Specificity            0.9701  0.89151   0.9693   0.9370   0.9383
## Pos Pred Value         0.9271  0.51624   0.8655   0.6753   0.7190
## Neg Pred Value         0.9387  0.95407   0.8957   0.9455   0.9631
## Prevalence             0.3076  0.13696   0.2370   0.1562   0.1623
## Detection Rate         0.2637  0.09992   0.1509   0.1106   0.1322
## Detection Prevalence   0.2845  0.19354   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9138  0.81052   0.8029   0.8227   0.8765
```

the accuracy of using the following method is 73.76 %

Usuing random forest method

```
sub_train$classe <- as.factor(sub_train$classe)
model2 <- randomForest(classe ~. , data = sub_train, method="class")
prediction2 <- predict(model2, sub_test, type = "class")
```

Testing the results using confusion matrix function:

```
confusionMatrix(prediction2, sub_test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    3    0    0    0
##          B    0 1132    4    0    0
##          C    0    4 1022    9    4
##          D    0    0    0  955    4
##          E    0    0    0    0 1074
##
## Overall Statistics
##
##                Accuracy : 0.9952
##                  95% CI : (0.9931, 0.9968)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.994
```

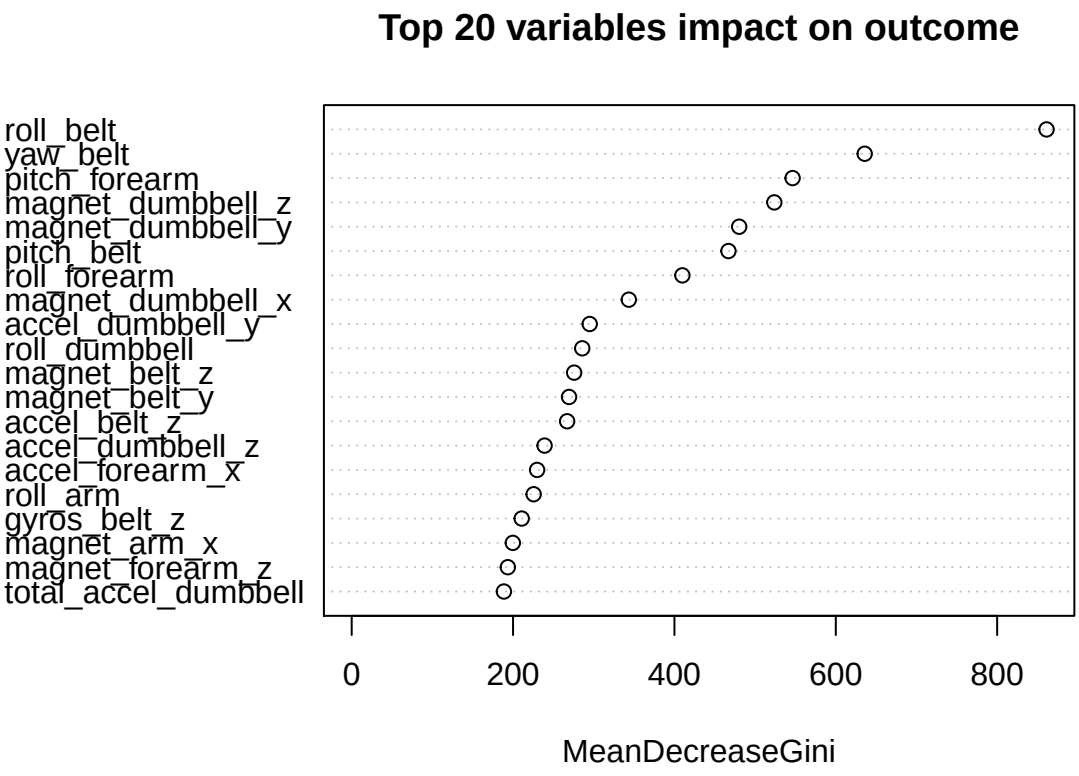
```
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9939   0.9961   0.9907   0.9926
## Specificity            0.9993   0.9992   0.9965   0.9992   1.0000
## Pos Pred Value         0.9982   0.9965   0.9836   0.9958   1.0000
## Neg Pred Value         1.0000   0.9985   0.9992   0.9982   0.9983
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1924   0.1737   0.1623   0.1825
## Detection Prevalence   0.2850   0.1930   0.1766   0.1630   0.1825
## Balanced Accuracy      0.9996   0.9965   0.9963   0.9949   0.9963
```

the accuracy of using the following method is 99.55 %

Based on the result the random forest method can be considered as a better method to use

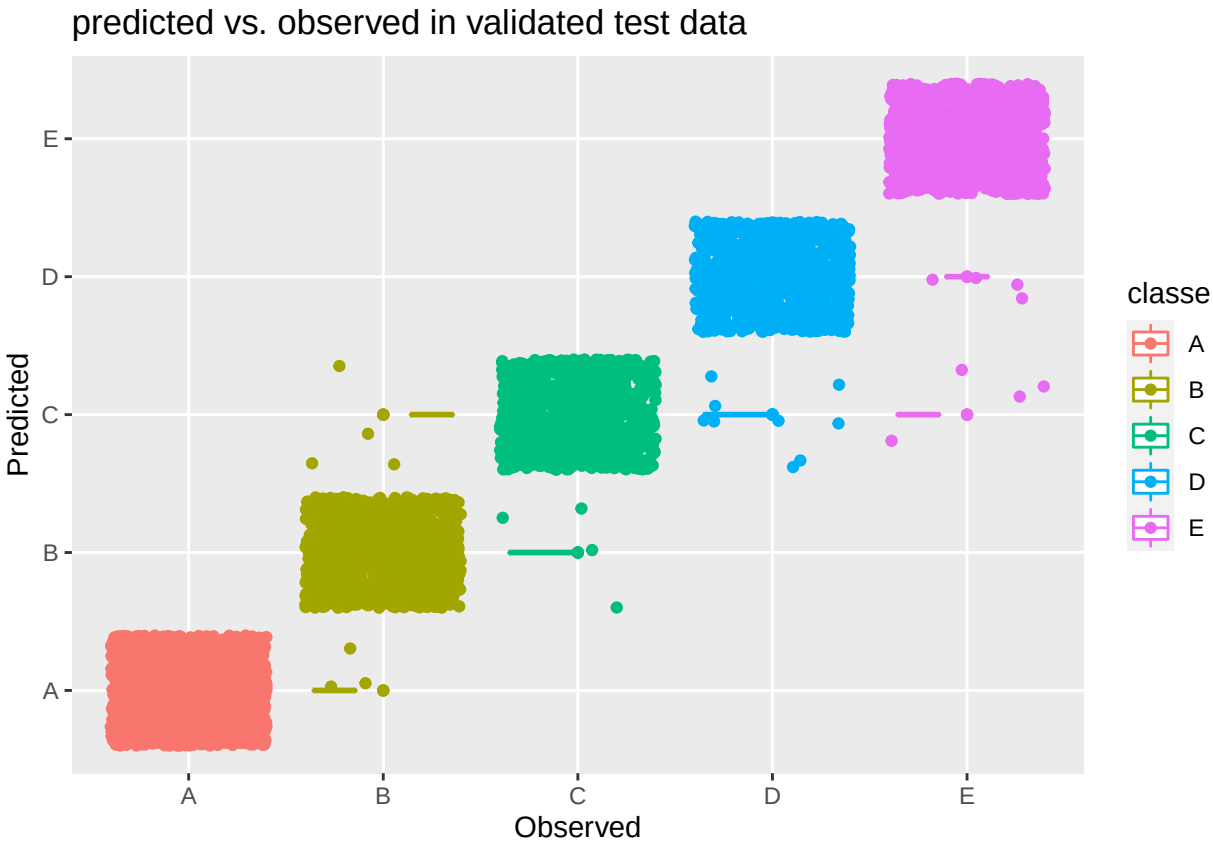
Bulding plot based on the results

```
varImpPlot(model2, n.var = 20, main = 'Top 20 variables impact on outcome')
```



```
res_plot <- qplot(classe, prediction2, data=sub_test,  colour= classe,
      main = "predicted vs. observed in validated test data", xlab = "Observed", ylab = "Predicted") +
  geom_boxplot()+ geom_jitter()

res_plot
```



Predicting final outcome levels on the original test data set using random forest algorithm

```
result <- predict(model2, test_data, type="class")
result
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```