Practical Machine Learning Course Project

José Maria Fernandes Marlet

2022-08-17

Executive Summary

What is the project:

Background Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

According to the reference: "Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E).

Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. Participants were supervised by an experienced weight lifter to make sure the execution complied to the manner they were supposed to simulate. The exercises were performed by six male participants aged between 20-28 years, with little weight lifting experience. We made sure that all participants could easily simulate the mistakes in a safe and controlled manner by using a relatively light dumbbell (1.25kg)."

Data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

What you should submit The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Main conclusions and considerations

1. The trainingClean dataset is splitted in two dataset, one for training other for testing, the partion will be 70% for training and 30% for testing.

2. result is a total of 53 covariates for the response variable "classe".

3. The models tested to correlate the 53 covariates with the response variable classe are: Random Forest (RF); Gradient boosted trees (GBT); Support Vector Machines (SVM).
4. The best model for this dataset is Random Forest with accuracy of 0.995.

Development of the Course Project - Practical Machine Learning

oading libraries and packages

```
library(lattice)

library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.1
```

```
library(kernlab)
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
library(rlang)
```

```
## Warning: package 'rlang' was built under R version 4.2.1
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.2.1
```

```
library(rpart)

library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.2.1
```

```
library(RColorBrewer)

library(rattle)
```

```
## Warning: package 'rattle' was built under R version 4.2.1
## Carregando pacotes exigidos: tibble
```

```
## Carregando pacotes exigidos: bitops
```

```
## 
## Attaching package: 'bitops'
```

```
## The following object is masked from 'package:rlang':
## 
##     %|%
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.2.1
```

```
## randomForest 4.7-1.1
```

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

```
## 
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:rattle':
## 
##     importance
```

```
## The following object is masked from 'package:ggplot2':
## 
##     margin
```

```r
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.2.1
```

```
## corrplot 0.92 loaded
```

```r
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 4.2.1
## Loaded gbm 2.1.8.1
```

```
set.seed(1234)
```

Loading data

The training and testing data sets come for the following url addresses: 1. Training: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

2. Testining: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The trainig data set has 19622 rows and 160 columns.

The testing data set has 20 rows and 160 columns.

From both data sets, the first seven columns don't bring useful information for the model's development and will be eliminated at the data cleaning step.

```
trainingSet <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")

dim(trainingSet)
```

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

```
testingSet <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")

dim(testingSet)
```

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

Data cleaning

```
trainingSetClean <- trainingSet[, colSums(is.na(trainingSet)) == 0]

testingSetClean <- testingSet[, colSums(is.na(testingSet)) == 0]

# Remotion from the first seven columns as they don't have impact on the outcome class

trainingClean <- trainingSetClean[, -c(1:7)]

testingClean <- testingSetClean[, -c(1:7)]

dim(trainingClean)
```

```
## [1] 19622    86
```

```
dim(testingClean)
```

```
## [1] 20 53
```

```
# Removing near zero variance variables: these variables are almost constant and dont't
# add value to the models.

nzv <- nearZeroVar(trainingClean)

trainingClean <- trainingClean[, -nzv]

dim(trainingClean)
```

```
## [1] 19622    53
```

```
# Converting "classe" to a factor

trainingClean$classe <- as.factor(trainingClean$classe)
```

Preparing the data sets for prediction

The trainingClean dataset is splitted in two dataset, one for training other for testing, the partion will be 70% for training and 30% for testing.

The result is a total of 53 covariates for the response variable “classe”.

```
inTrain <- createDataPartition(trainingClean$classe, p = 0.7, list = FALSE)

trainSet <- trainingClean[ inTrain,]

testSet <- trainingClean[ -inTrain,]

dim(trainSet)
```

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

```
dim(testSet)
```

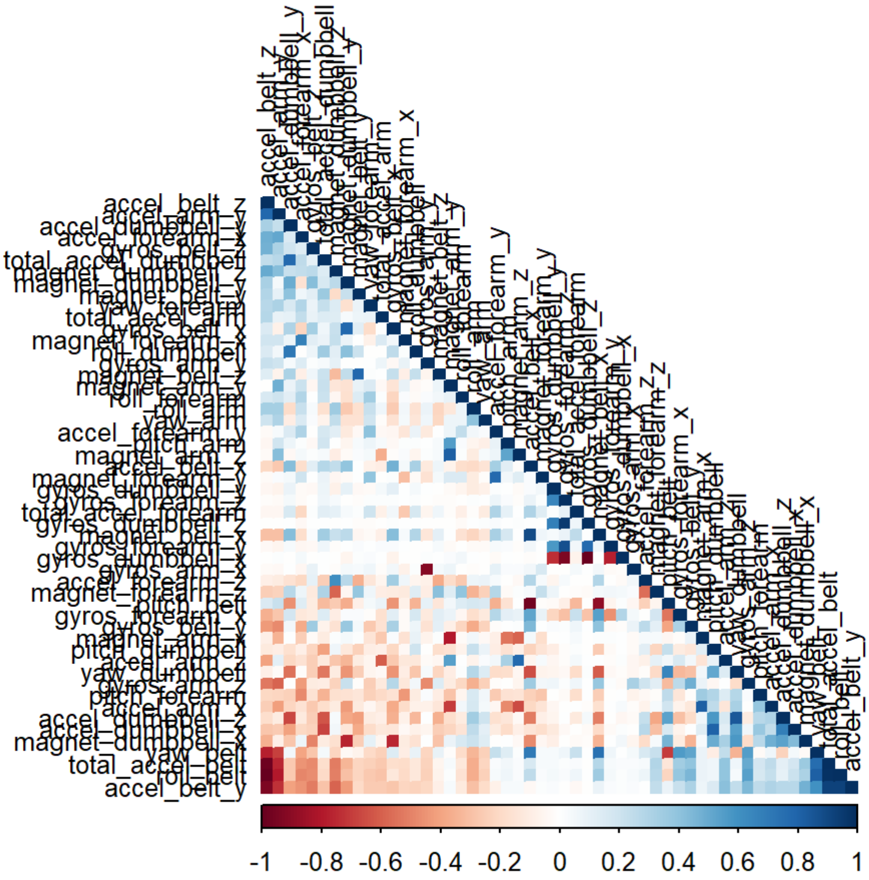
```
## [1] 5885   53
```

Correlation analysis

The correlation between covariates excluding the the response variable is showed below.The darker is the color the higher the covariates are correlated.

```
correlationMatrix <- cor(trainSet[, -53])

corrplot(correlationMatrix, method = "color", type = "lower", order = "FPC", tl.cex = 0.8, tl.c
ol = rgb(0, 0, 0))
```



Testing Models

The chosen models to be tested to correlate the 53 covariates with the response variable classe are:

1. Random Forest (RF)
2. Gradient boosted trees (GBT)
3. Support Vector Machines (SVM)

The 3 models are compared for accuracy of the predictions.

```
trainCtrl <- trainControl(method = "cv", number = 3, verboseIter = FALSE)
```

Random Forest (RF)

Shows Accuracy of 0.995.

```
mod_rf <- train(classe ~ ., data = trainSet, method = "rf", trControl = trainCtrl, tuneLength =
5)

pred_rf <- predict(mod_rf, newdata = testSet)

cm_rf <- confusionMatrix(pred_rf, testSet$classe)

cm_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    3    0    0    0
##          B    1 1132   11    0    0
##          C    0    4 1014    6    0
##          D    0    0    1  957    0
##          E    0    0    0    1 1082
##
## Overall Statistics
##
##                Accuracy : 0.9954
##                  95% CI : (0.9933, 0.997)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9942
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9939   0.9883   0.9927   1.0000
## Specificity            0.9993   0.9975   0.9979   0.9998   0.9998
## Pos Pred Value         0.9982   0.9895   0.9902   0.9990   0.9991
## Neg Pred Value         0.9998   0.9985   0.9975   0.9986   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1924   0.1723   0.1626   0.1839
## Detection Prevalence   0.2848   0.1944   0.1740   0.1628   0.1840
## Balanced Accuracy      0.9993   0.9957   0.9931   0.9963   0.9999
```

Gradient boosted trees (GBT)

Shows accuracy of 0.993.

```
mod_gbm <- train(classe ~ ., data = trainSet, method = "gbm", trControl = trainCtrl, tuneLength
= 5, verbose = F)

pred_gbm <- predict(mod_gbm, newdata = testSet)

cm_gbm <- confusionMatrix(pred_gbm, testSet$classe)

cm_gbm
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    5    0    0    0
##          B    2 1128    9    0    0
##          C    0    6 1014    9    2
##          D    0    0    3  952    0
##          E    0    0    0    3 1080
## 
## Overall Statistics
## 
##                Accuracy : 0.9934
##                  95% CI : (0.991, 0.9953)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9916
## 
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9988   0.9903   0.9883   0.9876   0.9982
## Specificity            0.9988   0.9977   0.9965   0.9994   0.9994
## Pos Pred Value         0.9970   0.9903   0.9835   0.9969   0.9972
## Neg Pred Value         0.9995   0.9977   0.9975   0.9976   0.9996
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2841   0.1917   0.1723   0.1618   0.1835
## Detection Prevalence   0.2850   0.1935   0.1752   0.1623   0.1840
## Balanced Accuracy      0.9988   0.9940   0.9924   0.9935   0.9988
```

Support Vector Machine (SVM)

Shows accuracy of 0.781.

```
mod_svm <- train(classe ~ ., data = trainSet, method = "svmLinear", trControl = trainCtrl, tuneLength = 5, verbose = F)

pred_svm <- predict(mod_svm, testSet)

cm_svm <- confusionMatrix(pred_svm, testSet$classe)

cm_svm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1537  154   79   69   50
##          B   29  806   90   46  152
##          C   40   81  797  114   69
##          D   61   22   32  697   50
##          E    7   76   28   38  761
##
## Overall Statistics
##
##                Accuracy : 0.7813
##                  95% CI : (0.7705, 0.7918)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.722
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9182   0.7076   0.7768   0.7230   0.7033
## Specificity            0.9164   0.9332   0.9374   0.9665   0.9690
## Pos Pred Value         0.8137   0.7177   0.7239   0.8086   0.8363
## Neg Pred Value         0.9657   0.9301   0.9521   0.9468   0.9355
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2612   0.1370   0.1354   0.1184   0.1293
## Detection Prevalence   0.3210   0.1908   0.1871   0.1465   0.1546
## Balanced Accuracy      0.9173   0.8204   0.8571   0.8447   0.8362
```

Model choice:

The model that best fits the data set is the Random Forest.

Predictions on test set

```
pred <- predict(mod_rf, testingClean)

print(pred)
```

```
##  [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
```

References

1. https://rpubs.com/bzhang93/coursera-machine-learning-project
2. https://rpubs.com/Marcela/practicalmachinelearning
3. https://rpubs.com/mchenini/353207

4. https://rpubs.com/EsosaOrumwese/835452
5. https://rpubs.com/neerajkbit/pmlproject
6. https://rpubs.com/vinsanity195/895801