Human Activity Quality in Weight Lifting Exercise

Klever Mera

Overview

This project consists on predicting the Quality of Human Activity when a person develops weight lifting exercises. The datasets were obtained considering 6 participants who performed the exercises correctly and incorrectly in 5 different ways.

Loading and Processing the Raw Data

First, training and testing csv files will be downloadad, and then work with the training file to create both the training dataset and the test dataset respectivly. The training partition will be 75% of the entire dataset and 25% to the test part which will be done by createDataPartition command included in the 'caret' package. The testing csv file won't be used till the time to work with the test case and the seed was set to 998.

Training

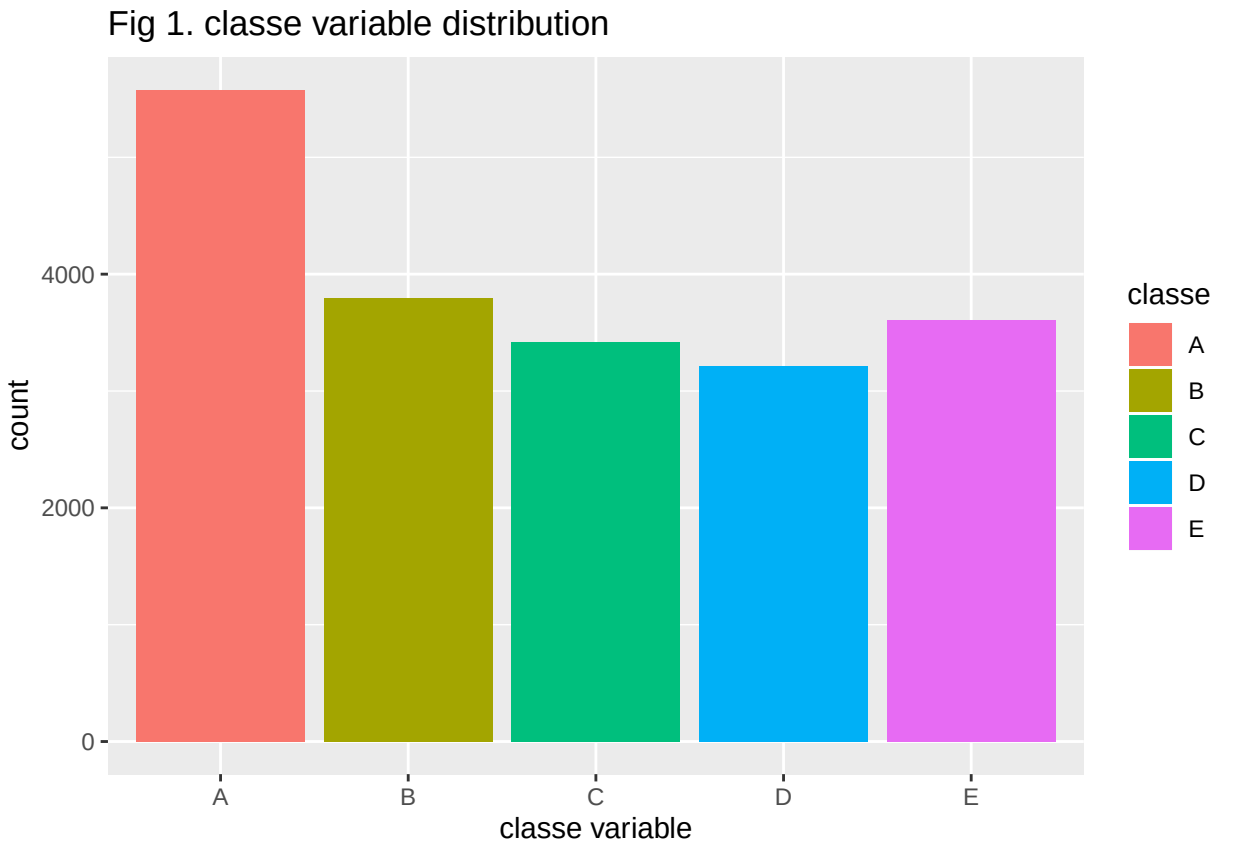
Training csv file has 19622 observations and 160 variables. The 'classe' variable is the one that will be used as a predicted one, and this variable has 5 levels, which are: "exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (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." Read more: http://groupware.les.inf.puc-rio.br/har#ixzz5u2aO6IXV

```
dim(train_file)
```

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

The Figure 1 shows the distribution of the 5 Classes (A, B, C, D, and E) of the classe variable in the training csv file. Class A has more quantity, which means the participants were doing the exercise in the correct way.

```
fig1
```



The first review of the training dataset is the proportion of the missing values:

```
percent(mean(is.na(training)))
```

```
## [1] "61.3%"
```

The proportion of the missing values is high (60%+), so a strategy is needed to deal with. In this case, the columns with high rate of missing values won't be considered. Besides, the first seven columns didn't provide information to take into account for prediction, so those columns will also be eliminated.

The new proportion of missing values is really low and the new data set has 53 columns now, those variables (except classe) will be considered as predictors.

```
percent(mean(is.na(new_training)))
```

```
## [1] "0%"
```

```
dim(new_training)
```

```
## [1] 14718    53
```

Testing

The same approaches considering for the training dataset will be applied for the testing dataset, so the same strategy for missing values and for the first seven columns.

The training dataset has been cleaned.

```
percent(mean(is.na(new_testing)))
```

```
## [1] "0%"
```

```
dim(new_testing)
```

```
## [1] 4904   53
```

Modeling

Based on the power of Random Forest as an ensembling machine learning algorithm this is the model that will be used to predict, so in the train command of caret package the method will be "rf". But before that, some tuning parameters should be considered such as type of resampling, in this case repeated (10 times) 10-fold cross-validation has been defined for assessing model performance; moreover, and in order to speed up the process, the option mtry() was considered, which basically refers to the number of variables available for splitting at each tree node, so in this case mtry was set to 7. Finally, paralell processing was also considered to improve Random Forest performance.

```
cluster <- makeCluster(detectCores() - 1)
registerDoParallel(cluster)

fitControlRF <- trainControl(method = "repeatedcv",
                             number = 10,
                             repeats = 10,
                             allowParallel = TRUE)

modFitRF <- train(classe ~ ., data = new_training, method = "rf",
                  trControl = fitControlRF,
                          tuneGrid=data.frame(mtry = 7))

stopCluster(cluster)
registerDoSEQ()
print(modFitRF)
```

```
## Random Forest
##
## 14718 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 13245, 13245, 13246, 13248, 13247, 13246, ...
## Resampling results:
##
```

```
##   Accuracy   Kappa
##   0.9947818  0.9933992
##
## Tuning parameter 'mtry' was held constant at a value of 7
```

Confusion Matrix

The confusion matrix is useful to compare model performance. If the diagonal values are high and all other values are low, the desired classes are being predicted correctly.

```
predRF <- predict(modFitRF, new_testing)
confmatRF <- confusionMatrix(predRF, new_testing$classe)
confmatRF
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    6    0    0    0
##          B    0  942    3    0    0
##          C    0    1  852    6    0
##          D    0    0    0  798    1
##          E    0    0    0    0  900
##
## Overall Statistics
##
##                Accuracy : 0.9965
##                  95% CI : (0.9945, 0.998)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9956
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9926   0.9965   0.9925   0.9989
## Specificity            0.9983   0.9992   0.9983   0.9998   1.0000
## Pos Pred Value         0.9957   0.9968   0.9919   0.9987   1.0000
## Neg Pred Value         1.0000   0.9982   0.9993   0.9985   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1921   0.1737   0.1627   0.1835
## Detection Prevalence   0.2857   0.1927   0.1752   0.1629   0.1835
## Balanced Accuracy      0.9991   0.9959   0.9974   0.9961   0.9994
```

As we can see, the diagonal values are high and just a few and low false positives and false negatives, so the model has done a good work with the prediction. Furthermore, the Accuracy of the model is 0.9947818, so expected out of sample error is 0.0052.

Test case

Finally, it's time to use the testing csv file in order to test the model. This file has only 20 observations and the variable classe isn't present. As was done for training and testing dataset, the strategy to manage missing values will be applied and also the first seven columns will be eliminated.

```
predRF2 <- predict(modFitRF, testing20)
testing20$classe <- predRF2
testing20$classe
```

```
##  [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
```

Conclusions

In the raw data there were plenty of missing values, so a procedure to manage them was considered and in that way many columns were eliminated. Besides, the first seven columns were eliminated because the information, in this case, wasn't useful for the prediction model, for instance 'user_name'. Pre-procesing options such as PCA wasn't also considered.

Random Forest was the only prediction model considered because during the definition model was also considered Boosting but the accuracy was low and when the model combination was applied the accuracy was the same as the Random Forest, so only this model was used.

Random Forest method ("rf") in the caret package consumes a lot of computer resources, so parallel processing technique was also included in the model in order to speed up the process.

The confusion matrix shows the accuracy which in this case is 0.9965. The Precision/Recall and F1 Score could also be included in a next analysis as a key metrics for a complete model evaluation.