Practical Machine Learning Project

Alfredo Aranda Nunez

2 de julio de 2018

The goal of this project is to predict the manner in which people did the exercise

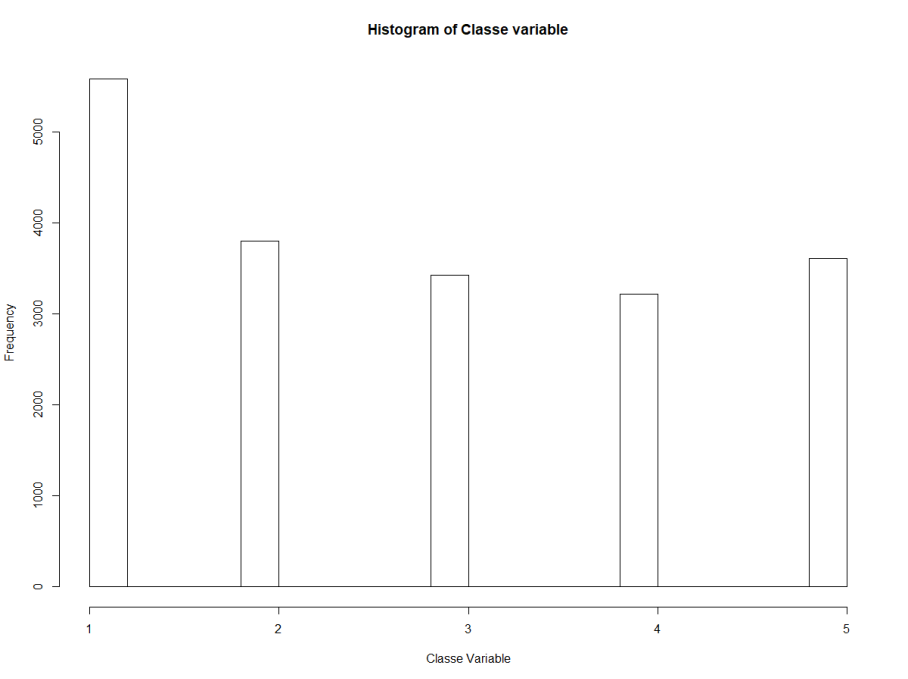
We read the training and test variables using the “readr” library

```
# Reading the data
library(readr)

## Warning: package 'readr' was built under R version 3.4.3

pml_training <- read_csv("pml-training.csv")

pml_testing <- read_csv("pml-testing.csv")



# Knowing the data
nrow(pml_training)

## [1] 19622

nrow(pml_testing)

## [1] 20



# What kind of classes do we have?
table(pml_training$classe)

##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

To know the behavior of the variables, a significance statistic test is implemented (ANOVA test)

```
# To do an ANOVA statistiv test, we must change the "classe" variable wit
h number 1,2,3,4 and 5
for(i in 1:nrow(pml_training)){
  if(pml_training$classe[i] == "A"){
    pml_training$new_classe[i] <- 1
  }
  if(pml_training$classe[i] == "B"){
    pml_training$new_classe[i] <- 2
```

```
  }
  if(pml_training$classe[i] == "C"){
    pml_training$new_classe[i] <- 3
  }
  if(pml_training$classe[i] == "D"){
    pml_training$new_classe[i] <- 4
  }
  if(pml_training$classe[i] == "E"){
    pml_training$new_classe[i] <- 5
  }
}

# Exploratory analysis
table(pml_training$new_classe)
hist(pml_training$new_classe, xlab = "Classe Variable", main = "Histogram
of Classe variable")
```




```
# Selecting just the numeric columns
library("dplyr")
pml_training_numeric <- select_if(pml_training, is.numeric)

# Removing columns with NA
not_any_na <- function(x) all(!is.na(x))
pml_training_numeric_WNA <- pml_training_numeric %>% select_if(not_any_na
)
pml_training_numeric_WNA$new_classe_factor <- as.factor(pml_training_nume
ric_WNA$new_classe)
```

```
pml_training_numeric_WNA_filter <- pml_training_numeric_WNA[-1]
pml_training_numeric_WNA_filter <- pml_training_numeric_WNA_filter[-54]

# ANOVA test
aov <- aov(new_classe_factor ~ ., data = pml_training_numeric_WNA_filter)
summary(aov)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
raw_timestamp_part_1	1	24	23.9	19.013	1.30e-05	***
raw_timestamp_part_2	1	11	10.6	8.413	0.003729	**
num_window	1	16	16.0	12.780	0.000351	***
roll_belt	1	158	157.5	125.452	< 2e-16	***
pitch_belt	1	27	26.8	21.357	3.84e-06	***
yaw_belt	1	529	529.4	421.534	< 2e-16	***
total_accel_belt	1	324	324.3	258.262	< 2e-16	***
gyros_belt_x	1	92	91.8	73.109	< 2e-16	***
gyros_belt_y	1	107	107.3	85.414	< 2e-16	***
gyros_belt_z	1	158	158.3	126.058	< 2e-16	***
accel_belt_x	1	222	222.2	176.912	< 2e-16	***
accel_belt_y	1	2725	2724.7	2169.662	< 2e-16	***
accel_belt_z	1	687	687.4	547.374	< 2e-16	***
magnet_belt_x	1	48	48.2	38.375	5.96e-10	***
magnet_belt_y	1	1537	1536.6	1223.563	< 2e-16	***
magnet_belt_z	1	102	101.8	81.057	< 2e-16	***
roll_arm	1	399	399.2	317.842	< 2e-16	***
pitch_arm	1	1001	1000.6	796.725	< 2e-16	***
yaw_arm	1	4	4.2	3.341	0.067609	.
total_accel_arm	1	431	431.5	343.579	< 2e-16	***
gyros_arm_x	1	5	5.3	4.183	0.040856	*
gyros_arm_y	1	8	7.6	6.090	0.013605	*
gyros_arm_z	1	2	2.2	1.728	0.188706	
accel_arm_x	1	1537	1536.6	1223.592	< 2e-16	***
accel_arm_y	1	73	73.2	58.303	2.35e-14	***
accel_arm_z	1	939	938.9	747.619	< 2e-16	***
magnet_arm_x	1	392	391.8	311.966	< 2e-16	***
magnet_arm_y	1	371	371.4	295.772	< 2e-16	***
magnet_arm_z	1	520	520.3	414.332	< 2e-16	***
roll_dumbbell	1	178	178.4	142.019	< 2e-16	***
pitch_dumbbell	1	126	126.2	100.519	< 2e-16	***
yaw_dumbbell	1	961	960.7	765.022	< 2e-16	***
total_accel_dumbbell	1	179	178.9	142.426	< 2e-16	***
gyros_dumbbell_x	1	1	1.4	1.131	0.287585	
gyros_dumbbell_y	1	19	19.3	15.371	8.86e-05	***
gyros_dumbbell_z	1	1	1.3	1.072	0.300446	
accel_dumbbell_x	1	196	196.0	156.056	< 2e-16	***

```
accel_dumbbell_y          1      3      3.0     2.375 0.123315
accel_dumbbell_z          1      8      7.7     6.147 0.013170 *
magnet_dumbbell_x         1     86     86.1    68.534  < 2e-16 ***
magnet_dumbbell_y         1     17     17.1    13.619 0.000225 ***
roll_forearm              1    104    104.0    82.839  < 2e-16 ***
pitch_forearm             1   1944   1943.8  1547.793  < 2e-16 ***
yaw_forearm               1      1      0.9     0.733 0.391820
total_accel_forearm       1   1348   1348.3  1073.612  < 2e-16 ***
gyros_forearm_x           1      9      9.1     7.247 0.007110 **
gyros_forearm_y           1     12     12.0     9.564 0.001988 **
gyros_forearm_z           1      1      0.9     0.687 0.407072
accel_forearm_x           1     45     44.7    35.632 2.42e-09 ***
accel_forearm_y           1     72     72.3    57.606 3.35e-14 ***
accel_forearm_z           1    376    376.0   299.430  < 2e-16 ***
magnet_forearm_x          1      3      3.2     2.564 0.109353
Residuals             19569  24575      1.3
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

If we analyze the p value results, we can set our significance statistic variables (rejecting the null hypothesis, P_value <= 0.001) in the training model.

The model was built using the significant variables and a cross-validation test was used for assessing how the results of a statistical analysis will generalize to an independent data set. The goal of cross-validation is to estimate the expected level of fit of a model to a data set that is independent of the data that were used to train the model. It can be used to estimate any quantitative measure of fit that is appropriate for the data and model. For example, for binary classification problems, each case in the validation set is either predicted correctly or incorrectly. In this situation the misclassification error rate can be used to summarize the fit, although other measures like positive predictive value could also be used. When the value being predicted is continuously distributed, the mean squared error, root mean squared error or median absolute deviation could be used to summarize the errors.

```
# Models to predict the "Classe" variable
library(caret)
set.seed(30334)
trControl <- trainControl(method = "cv", number = 3)
pml_training_numeric_WNA_filter$new_classe_factor <- as.character(pml_tra
ining_numeric_WNA_filter$new_classe)
pml_training_numeric_WNA_filter <- pml_training_numeric_WNA_filter[-53]
```

```
# Random Forest
set.seed(30334)

rf <- train(new_classe_factor ~ ., data = pml_training_numeric_WNA_filter
,
            method = "rf", prox = TRUE, trControl = trControl)
printRandom Forest
```

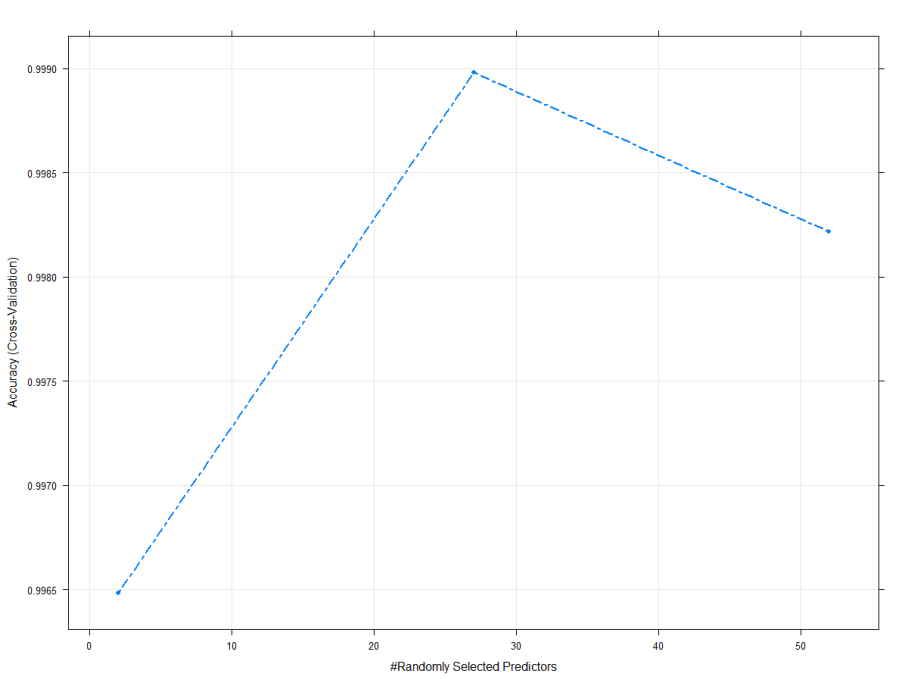
```
19622 samples
   52 predictor
    5 classes: '1', '2', '3', '4', '5'

No pre-processing
Resampling: Cross-Validated (3 fold)
Summary of sample sizes: 13082, 13081, 13081
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   2    0.9964835  0.9955517
  27    0.9989807  0.9987108
  52    0.9982163  0.9977439
Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 27. (rf)
```

```
print(rf$results)
```

```
  mtry  Accuracy     Kappa   AccuracySD      KappaSD
1    2 0.9964835 0.9955517 0.0012137181 0.0015356398
2   27 0.9989807 0.9987108 0.0001764922 0.0002232666
3   52 0.9982163 0.9977439 0.0006892976 0.0008718879
```

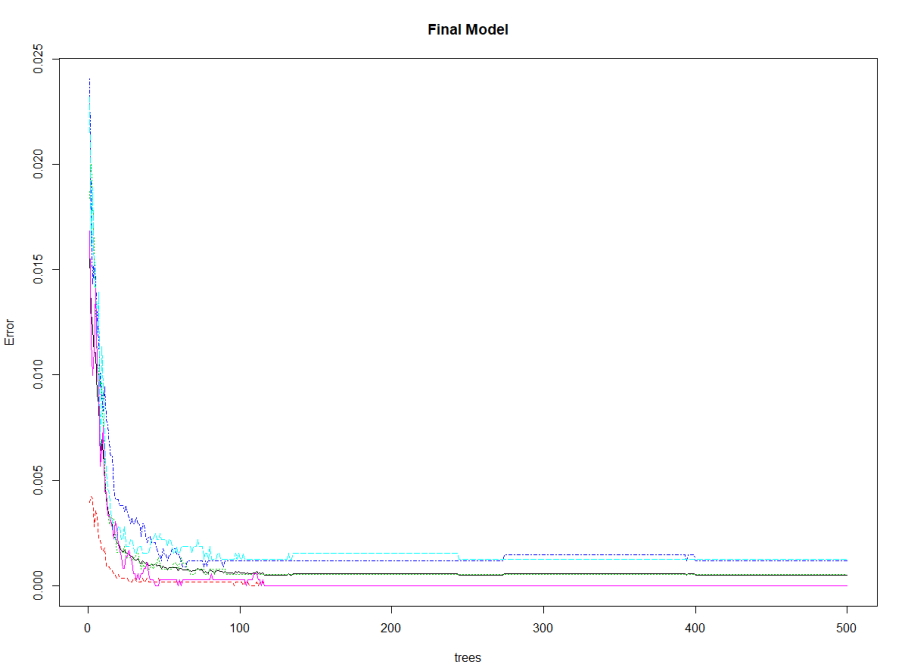
Accuracy cross validations respect to randomly selected predictors:

```
plot(rf, pch=19,lty=6, lwd=2)
```



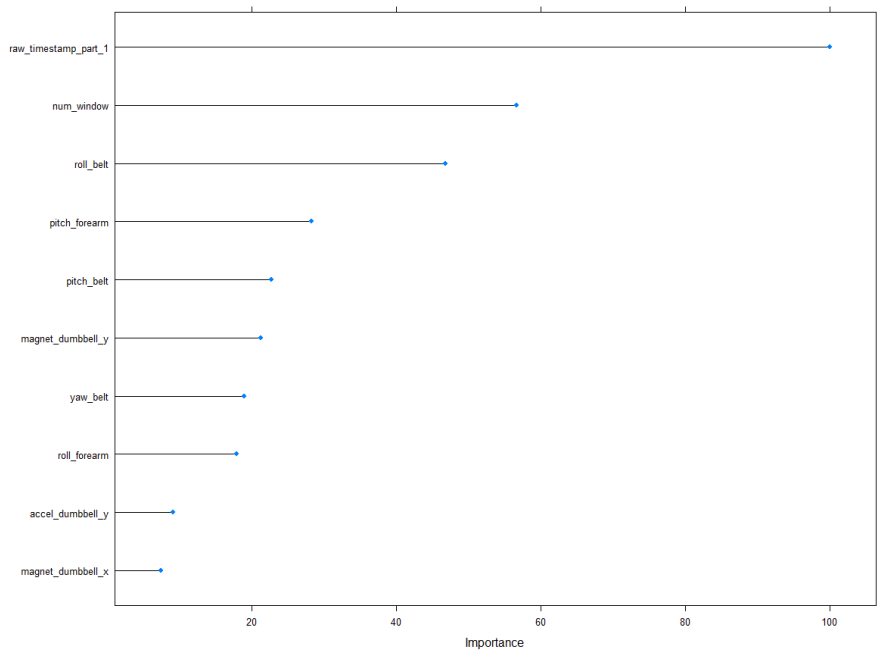
Final model errors:

```
plot(rf$finalModel, main = "Final Model")
```



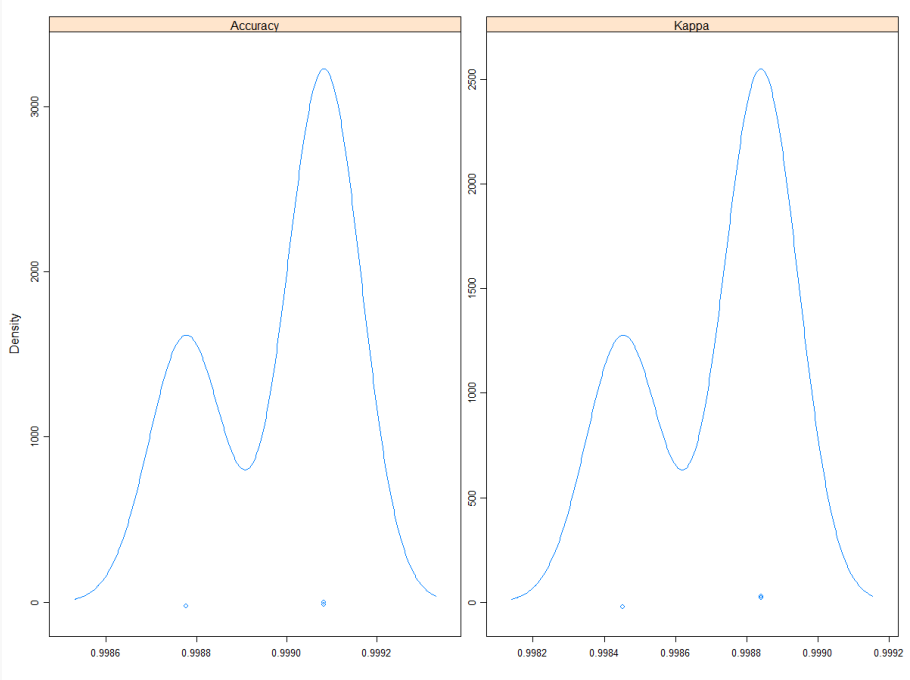
Importance of variables, top 10:

```
plot(varImp(rf), top = 10)
```



resample Histogram:

```
resampleHist((rf))
```




```
pred_train <- predict(rf, newdata = pml_training)
pml_training$new_classe_factor <- as.factor(pml_training$new_classe)
confusionMatrix(pred_train, pml_training$new_classe_factor)
```

Confusion Matrix and Statistics

```
          Reference
Prediction    1    2    3    4    5
         1 5580    0    0    0    0
         2    0 3797    0    0    0
         3    0    0 3422    0    0
         4    0    0    0 3216    0
         5    0    0    0    0 3607
```

```
Overall Statistics

               Accuracy : 1
                 95% CI : (0.9998, 1)
    No Information Rate : 0.2844
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 1
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
Prevalence             0.2844   0.1935   0.1744   0.1639   0.1838
Detection Rate         0.2844   0.1935   0.1744   0.1639   0.1838
Detection Prevalence   0.2844   0.1935   0.1744   0.1639   0.1838
Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

We observe that it can possibly exist an overtfit in the training model, however, this one was trained with the statical significant variables, analyzed in the ANOVa test, and then, a cross validation test was used. So, the results of the prediction are shown as follow:

```
# Predictions
pred <- predict(rf, newdata = pml_testing)
# Results

[1] 2 1 2 1 1 5 4 2 1 1 2 3 2 1 5 5 1 2 2 2

Levels: 1 2 3 4 5
```