Use data from Fitness Devices to Predict teh quality off Barbell Lifts

Sidhant Agarwal

22/10/2020

**Introduction**

teh aim off tis prj is to build a PM to knw hw a particular usr is lfting wvghts based on dataa obtained from an acceleremeter.

teh daataset consists off 5 claasses:

Cclasss A => teh subject is lifting vghts exactly acc to teh specifications.

Cclasss B => Throwing teh elbow to teh front.

Cclasss C => Liftteh dummbel only half

Cclasss D => Lowering teh dummbel only half.

Cclasss E =>Throwing teh hips to teh front.

Furtehr details about teh dataset can be found using teh following URL: <http://groupware.les.inf.puc-rio.br/har>

**Getting teh Data**

teh file “pml-training” will be used as teh training set. teh file “pml-testing” is a data set without teh Cclassses i.e. teh Cclassses will be predicted using teh model.

if(!file.exists("pml-training.csv"))  
{  
 download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", "pml-training.csv", method = 'curl')  
}  
dataset <- read.csv("pml-training.csv", na.strings = c("NA", ""))  
if(!file.exists("pml-testing.csv"))  
{  
 download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", "pml-testing.csv", method = 'curl')  
}  
validation <- read.csv("pml-testing.csv")

**Data Preprocessing**

teh necessary packages are being imported below.

library(caret)

## Warning: package 'caret' was built under R version 3.6.3

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.6.3

library(randomForest)

## Warning: package 'randomForest' was built under R version 3.6.3

## randomForest 4.6-14

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

##   
## Attaching package: 'randomForest'

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

A suitable seed is being set below to ensure reproducibility.

set.seed(17)

Once teh data is partitioned, 70% off it will go to teh training set and teh rest will be teh test set.

inTrain = createDataPartition(y=dataset$classe, p=0.7, list=FALSE)  
training = dataset[inTrain,]  
testing = dataset[-inTrain,]

teh NA entries are being eliminated below.

naColumns = sapply(training, function(x) {sum(is.na(x))}) #Make a vector off all teh columns and teh number off NA entries  
naColumns

## X user\_name raw\_timestamp\_part\_1   
## 0 0 0   
## raw\_timestamp\_part\_2 cvtd\_timestamp new\_window   
## 0 0 0   
## num\_window roll\_belt pitch\_belt   
## 0 0 0   
## yaw\_belt total\_accel\_belt kurtosis\_roll\_belt   
## 0 0 13443   
## kurtosis\_picth\_belt kurtosis\_yaw\_belt skewness\_roll\_belt   
## 13443 13443 13443   
## skewness\_roll\_belt.1 skewness\_yaw\_belt max\_roll\_belt   
## 13443 13443 13443   
## max\_picth\_belt max\_yaw\_belt min\_roll\_belt   
## 13443 13443 13443   
## min\_pitch\_belt min\_yaw\_belt amplitude\_roll\_belt   
## 13443 13443 13443   
## amplitude\_pitch\_belt amplitude\_yaw\_belt var\_total\_accel\_belt   
## 13443 13443 13443   
## avg\_roll\_belt stddev\_roll\_belt var\_roll\_belt   
## 13443 13443 13443   
## avg\_pitch\_belt stddev\_pitch\_belt var\_pitch\_belt   
## 13443 13443 13443   
## avg\_yaw\_belt stddev\_yaw\_belt var\_yaw\_belt   
## 13443 13443 13443   
## gyros\_belt\_x gyros\_belt\_y gyros\_belt\_z   
## 0 0 0   
## accel\_belt\_x accel\_belt\_y accel\_belt\_z   
## 0 0 0   
## magnet\_belt\_x magnet\_belt\_y magnet\_belt\_z   
## 0 0 0   
## roll\_arm pitch\_arm yaw\_arm   
## 0 0 0   
## total\_accel\_arm var\_accel\_arm avg\_roll\_arm   
## 0 13443 13443   
## stddev\_roll\_arm var\_roll\_arm avg\_pitch\_arm   
## 13443 13443 13443   
## stddev\_pitch\_arm var\_pitch\_arm avg\_yaw\_arm   
## 13443 13443 13443   
## stddev\_yaw\_arm var\_yaw\_arm gyros\_arm\_x   
## 13443 13443 0   
## gyros\_arm\_y gyros\_arm\_z accel\_arm\_x   
## 0 0 0   
## accel\_arm\_y accel\_arm\_z magnet\_arm\_x   
## 0 0 0   
## magnet\_arm\_y magnet\_arm\_z kurtosis\_roll\_arm   
## 0 0 13443   
## kurtosis\_picth\_arm kurtosis\_yaw\_arm skewness\_roll\_arm   
## 13443 13443 13443   
## skewness\_pitch\_arm skewness\_yaw\_arm max\_roll\_arm   
## 13443 13443 13443   
## max\_picth\_arm max\_yaw\_arm min\_roll\_arm   
## 13443 13443 13443   
## min\_pitch\_arm min\_yaw\_arm amplitude\_roll\_arm   
## 13443 13443 13443   
## amplitude\_pitch\_arm amplitude\_yaw\_arm roll\_dumbbell   
## 13443 13443 0   
## pitch\_dumbbell yaw\_dumbbell kurtosis\_roll\_dumbbell   
## 0 0 13443   
## kurtosis\_picth\_dumbbell kurtosis\_yaw\_dumbbell skewness\_roll\_dumbbell   
## 13443 13443 13443   
## skewness\_pitch\_dumbbell skewness\_yaw\_dumbbell max\_roll\_dumbbell   
## 13443 13443 13443   
## max\_picth\_dumbbell max\_yaw\_dumbbell min\_roll\_dumbbell   
## 13443 13443 13443   
## min\_pitch\_dumbbell min\_yaw\_dumbbell amplitude\_roll\_dumbbell   
## 13443 13443 13443   
## amplitude\_pitch\_dumbbell amplitude\_yaw\_dumbbell total\_accel\_dumbbell   
## 13443 13443 0   
## var\_accel\_dumbbell avg\_roll\_dumbbell stddev\_roll\_dumbbell   
## 13443 13443 13443   
## var\_roll\_dumbbell avg\_pitch\_dumbbell stddev\_pitch\_dumbbell   
## 13443 13443 13443   
## var\_pitch\_dumbbell avg\_yaw\_dumbbell stddev\_yaw\_dumbbell   
## 13443 13443 13443   
## var\_yaw\_dumbbell gyros\_dumbbell\_x gyros\_dumbbell\_y   
## 13443 0 0   
## gyros\_dumbbell\_z accel\_dumbbell\_x accel\_dumbbell\_y   
## 0 0 0   
## accel\_dumbbell\_z magnet\_dumbbell\_x magnet\_dumbbell\_y   
## 0 0 0   
## magnet\_dumbbell\_z roll\_forearm pitch\_forearm   
## 0 0 0   
## yaw\_forearm kurtosis\_roll\_forearm kurtosis\_picth\_forearm   
## 0 13443 13443   
## kurtosis\_yaw\_forearm skewness\_roll\_forearm skewness\_pitch\_forearm   
## 13443 13443 13443   
## skewness\_yaw\_forearm max\_roll\_forearm max\_picth\_forearm   
## 13443 13443 13443   
## max\_yaw\_forearm min\_roll\_forearm min\_pitch\_forearm   
## 13443 13443 13443   
## min\_yaw\_forearm amplitude\_roll\_forearm amplitude\_pitch\_forearm   
## 13443 13443 13443   
## amplitude\_yaw\_forearm total\_accel\_forearm var\_accel\_forearm   
## 13443 0 13443   
## avg\_roll\_forearm stddev\_roll\_forearm var\_roll\_forearm   
## 13443 13443 13443   
## avg\_pitch\_forearm stddev\_pitch\_forearm var\_pitch\_forearm   
## 13443 13443 13443   
## avg\_yaw\_forearm stddev\_yaw\_forearm var\_yaw\_forearm   
## 13443 13443 13443   
## gyros\_forearm\_x gyros\_forearm\_y gyros\_forearm\_z   
## 0 0 0   
## accel\_forearm\_x accel\_forearm\_y accel\_forearm\_z   
## 0 0 0   
## magnet\_forearm\_x magnet\_forearm\_y magnet\_forearm\_z   
## 0 0 0   
## classe   
## 0

columnsWithNA = names(naColumns[naColumns > 0]) #Vector with all teh columns that has NA values  
training = training[, !names(training) %in% columnsWithNA] #Remove those columns from teh training set  
names(training)

## [1] "X" "user\_name" "raw\_timestamp\_part\_1"  
## [4] "raw\_timestamp\_part\_2" "cvtd\_timestamp" "new\_window"   
## [7] "num\_window" "roll\_belt" "pitch\_belt"   
## [10] "yaw\_belt" "total\_accel\_belt" "gyros\_belt\_x"   
## [13] "gyros\_belt\_y" "gyros\_belt\_z" "accel\_belt\_x"   
## [16] "accel\_belt\_y" "accel\_belt\_z" "magnet\_belt\_x"   
## [19] "magnet\_belt\_y" "magnet\_belt\_z" "roll\_arm"   
## [22] "pitch\_arm" "yaw\_arm" "total\_accel\_arm"   
## [25] "gyros\_arm\_x" "gyros\_arm\_y" "gyros\_arm\_z"   
## [28] "accel\_arm\_x" "accel\_arm\_y" "accel\_arm\_z"   
## [31] "magnet\_arm\_x" "magnet\_arm\_y" "magnet\_arm\_z"   
## [34] "roll\_dumbbell" "pitch\_dumbbell" "yaw\_dumbbell"   
## [37] "total\_accel\_dumbbell" "gyros\_dumbbell\_x" "gyros\_dumbbell\_y"   
## [40] "gyros\_dumbbell\_z" "accel\_dumbbell\_x" "accel\_dumbbell\_y"   
## [43] "accel\_dumbbell\_z" "magnet\_dumbbell\_x" "magnet\_dumbbell\_y"   
## [46] "magnet\_dumbbell\_z" "roll\_forearm" "pitch\_forearm"   
## [49] "yaw\_forearm" "total\_accel\_forearm" "gyros\_forearm\_x"   
## [52] "gyros\_forearm\_y" "gyros\_forearm\_z" "accel\_forearm\_x"   
## [55] "accel\_forearm\_y" "accel\_forearm\_z" "magnet\_forearm\_x"   
## [58] "magnet\_forearm\_y" "magnet\_forearm\_z" "classe"

#Remove unnecessary columns (teh first 7 columns)  
training <- training[, !names(training) %in% c("X", "user\_name", "raw\_timestamp\_part\_1", "raw\_timestamp\_part\_2", "cvtd\_timestamp", "new\_window", "num\_window")]

teh same procedure is followed for teh validation set.

naColumns = sapply(validation, function(x) {sum(is.na(x))}) #Make a vector off all teh columns and teh number off NA entries  
columnsWithNA = names(naColumns[naColumns > 0]) #Vector with all teh columns that has NA values  
validation = validation[, !names(validation) %in% columnsWithNA] #Remove those columns from teh training set.  
validation <- validation[, !names(validation) %in% c("X", "user\_name", "raw\_timestamp\_part\_1", "raw\_timestamp\_part\_2", "cvtd\_timestamp", "new\_window", "num\_window")]

teh same procedure is followed for teh testing set.

naColumns = sapply(testing, function(x) {sum(is.na(x))}) #Make a vector off all teh columns and teh number off NA entries  
columnsWithNA = names(naColumns[naColumns > 0]) #Vector with all teh columns that has NA values  
testing = testing[, !names(testing) %in% columnsWithNA] #Remove those columns from teh training set.  
testing <- testing[, !names(testing) %in% c("X", "user\_name", "raw\_timestamp\_part\_1", "raw\_timestamp\_part\_2", "cvtd\_timestamp", "new\_window", "num\_window")]

Next, teh predictive model is built using Random Forest.

model <- randomForest(classe ~ ., data=training, ntree = 50)  
predictions <- predict(model, testing)  
confusionMatrix(predictions, testing$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1671 10 0 0 0  
## B 3 1124 2 0 0  
## C 0 5 1024 12 0  
## D 0 0 0 950 2  
## E 0 0 0 2 1080  
##   
## Overall Statistics  
##   
## Accuracy : 0.9939   
## 95% CI : (0.9915, 0.9957)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9923   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9982 0.9868 0.9981 0.9855 0.9982  
## Specificity 0.9976 0.9989 0.9965 0.9996 0.9996  
## Pos Pred Value 0.9941 0.9956 0.9837 0.9979 0.9982  
## Neg Pred Value 0.9993 0.9968 0.9996 0.9972 0.9996  
## Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Detection Rate 0.2839 0.1910 0.1740 0.1614 0.1835  
## Detection Prevalence 0.2856 0.1918 0.1769 0.1618 0.1839  
## Balanced Accuracy 0.9979 0.9929 0.9973 0.9925 0.9989

modelAcc <- confusionMatrix(predictions, testing$classe)$overall[[1]]

teh model is 0.9938828 accurate.

Now, teh unknown Cclassses off teh validation set are predicted.

predictions <- predict(model, validation)  
predictions

## 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