Machine Learning Weight lifting Exercise

Vijay Kumar Dubey

Monday, October 27, 2014

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

testdata<-read.csv(file.choose(),head=TRUE,stringsAsFactors=TRUE)  
str(testdata)

## 'data.frame': 20 obs. of 20 variables:  
## $ user\_name : Factor w/ 6 levels "adelmo","carlitos",..: 6 5 5 1 4 5 5 5 2 3 ...  
## $ new\_window : Factor w/ 1 level "no": 1 1 1 1 1 1 1 1 1 1 ...  
## $ num\_window : int 74 431 439 194 235 504 485 440 323 664 ...  
## $ roll\_belt : num 123 1.02 0.87 125 1.35 -5.92 1.2 0.43 0.93 114 ...  
## $ pitch\_belt : num 27 4.87 1.82 -41.6 3.33 1.59 4.44 4.15 6.72 22.4 ...  
## $ yaw\_belt : num -4.75 -88.9 -88.5 162 -88.6 -87.7 -87.3 -88.5 -93.7 -13.1 ...  
## $ total\_accel\_belt : int 20 4 5 17 3 4 4 4 4 18 ...  
## $ roll\_arm : num 40.7 0 0 -109 76.1 0 0 0 -137 -82.4 ...  
## $ pitch\_arm : num -27.8 0 0 55 2.76 0 0 0 11.2 -63.8 ...  
## $ yaw\_arm : num 178 0 0 -142 102 0 0 0 -167 -75.3 ...  
## $ total\_accel\_arm : int 10 38 44 25 29 14 15 22 34 32 ...  
## $ roll\_dumbbell : num -17.7 54.5 57.1 43.1 -101.4 ...  
## $ pitch\_dumbbell : num 25 -53.7 -51.4 -30 -53.4 ...  
## $ yaw\_dumbbell : num 126.2 -75.5 -75.2 -103.3 -14.2 ...  
## $ total\_accel\_dumbbell: int 9 31 29 18 4 29 29 29 3 2 ...  
## $ roll\_forearm : num 141 109 131 0 -176 150 155 -161 15.5 13.2 ...  
## $ pitch\_forearm : num 49.3 -17.6 -32.6 0 -2.16 1.46 34.5 43.6 -63.5 19.4 ...  
## $ yaw\_forearm : num 156 106 93 0 -47.9 89.7 152 -89.5 -139 -105 ...  
## $ total\_accel\_forearm : int 33 39 34 43 24 43 32 47 36 24 ...  
## $ problem\_id : int 1 2 3 4 5 6 7 8 9 10 ...

# Converting non numeric variable to numeric  
  
testdata$num\_window <- as.numeric(testdata$num\_window)  
  
testdata$total\_accel\_belt <- as.numeric(testdata$total\_accel\_belt)  
  
testdata$total\_accel\_arm <- as.numeric(testdata$total\_accel\_arm)  
  
testdata$total\_accel\_dumbbell <- as.numeric(testdata$total\_accel\_dumbbell)  
  
testdata$total\_accel\_forearm <- as.numeric(testdata$total\_accel\_forearm)  
  
testdata$problem\_id <- as.numeric(testdata$problem\_id)

# Training dataset preparation

WL<-read.csv(file.choose(),head=TRUE, na.strings=c("NA", "#DIV/0!"), stringsAsFactors=TRUE)  
  
names(WL)

## [1] "user\_name" "new\_window" "num\_window"   
## [4] "roll\_belt" "pitch\_belt" "yaw\_belt"   
## [7] "total\_accel\_belt" "roll\_arm" "pitch\_arm"   
## [10] "yaw\_arm" "total\_accel\_arm" "roll\_dumbbell"   
## [13] "pitch\_dumbbell" "yaw\_dumbbell" "total\_accel\_dumbbell"  
## [16] "roll\_forearm" "pitch\_forearm" "yaw\_forearm"   
## [19] "total\_accel\_forearm" "classe"

str(WL)

## 'data.frame': 19622 obs. of 20 variables:  
## $ user\_name : Factor w/ 6 levels "adelmo","carlitos",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ new\_window : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...  
## $ num\_window : int 11 11 11 12 12 12 12 12 12 12 ...  
## $ roll\_belt : num 1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...  
## $ pitch\_belt : num 8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...  
## $ yaw\_belt : num -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...  
## $ total\_accel\_belt : int 3 3 3 3 3 3 3 3 3 3 ...  
## $ roll\_arm : num -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...  
## $ pitch\_arm : num 22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...  
## $ yaw\_arm : num -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...  
## $ total\_accel\_arm : int 34 34 34 34 34 34 34 34 34 34 ...  
## $ roll\_dumbbell : num 13.1 13.1 12.9 13.4 13.4 ...  
## $ pitch\_dumbbell : num -70.5 -70.6 -70.3 -70.4 -70.4 ...  
## $ yaw\_dumbbell : num -84.9 -84.7 -85.1 -84.9 -84.9 ...  
## $ total\_accel\_dumbbell: int 37 37 37 37 37 37 37 37 37 37 ...  
## $ roll\_forearm : num 28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...  
## $ pitch\_forearm : num -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...  
## $ yaw\_forearm : num -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...  
## $ total\_accel\_forearm : int 36 36 36 36 36 36 36 36 36 36 ...  
## $ classe : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...

WL$num\_window <- as.numeric(WL$num\_window)  
  
WL$total\_accel\_belt <- as.numeric(WL$total\_accel\_belt)  
  
WL$total\_accel\_arm <- as.numeric(WL$total\_accel\_arm)  
  
WL$total\_accel\_dumbbell <- as.numeric(WL$total\_accel\_dumbbell)  
  
WL$total\_accel\_forearm <- as.numeric(WL$total\_accel\_forearm)

library(caret)

## Loading required package: lattice  
## Loading required package: ggplot2

# Further Splitting the "training dataset" into training dataset and testing dataset to perform CROSS VALIDATION

intrainCV <- createDataPartition(y=WL$classe, p=0.7, list = FALSE)  
  
trainingCV <- WL[intrainCV,]  
  
testingCV <- WL[-intrainCV,]  
  
dim(trainingCV); dim(testingCV)

## [1] 13737 20

## [1] 5885 20

# Training model using training dataset

trainmodlda<-train(classe~.,data = trainingCV,preProcess=c("center","scale"), method = "lda")

## Loading required package: MASS

# Testing model ussing the test dataset which was created after partitioning the training dataset

plda <- predict(trainmodlda,testingCV)

# Validation Predicting the outcome using the algorithim on the testing dataset

testplda <- predict(trainmodlda,testdata)

# Confusion matrix to test the out of sample error for the dataset that I have created for cross validation

confusionMatrix(testingCV$classe, plda)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1211 125 156 173 9  
## B 233 519 236 126 25  
## C 264 91 576 93 2  
## D 69 165 200 499 31  
## E 108 168 115 92 599  
##   
## Overall Statistics  
##   
## Accuracy : 0.5784   
## 95% CI : (0.5657, 0.5911)  
## No Information Rate : 0.3203   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.4647   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.6424 0.48596 0.44895 0.50763 0.8994  
## Specificity 0.8842 0.87129 0.90222 0.90514 0.9075  
## Pos Pred Value 0.7234 0.45566 0.56140 0.51763 0.5536  
## Neg Pred Value 0.8399 0.88432 0.85450 0.90165 0.9861  
## Prevalence 0.3203 0.18148 0.21801 0.16703 0.1132  
## Detection Rate 0.2058 0.08819 0.09788 0.08479 0.1018  
## Detection Prevalence 0.2845 0.19354 0.17434 0.16381 0.1839  
## Balanced Accuracy 0.7633 0.67862 0.67558 0.70639 0.9034