Practical Machine Learning Project Report

Bert

8 juli 2019

# Data Preprocessing

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

## Registered S3 methods overwritten by 'ggplot2':  
## method from   
## [.quosures rlang  
## c.quosures rlang  
## print.quosures rlang

library(rpart)  
library(rpart.plot)  
library(randomForest)

## 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

### Downloading the Data

trainUrl <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"  
testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"  
trainFile <- "./data/pml-training.csv"  
testFile <- "./data/pml-testing.csv"  
if (!file.exists("./data")) {  
 dir.create("./data")  
}  
if (!file.exists(trainFile)) {  
 download.file(trainUrl, destfile=trainFile)  
}  
if (!file.exists(testFile)) {  
 download.file(testUrl, destfile=testFile)  
}

### Reading the Data

After downloading the data from the data source, we can read the two csv files into two data frames.

trainRaw <- read.csv("./data/pml-training.csv")  
testRaw <- read.csv("./data/pml-testing.csv")  
dim(trainRaw)

## [1] 19622 160

#head(trainRaw)  
dim(testRaw)

## [1] 20 160

#head(testRaw)

The training data set has 19622 observations and 160 variables The testing data set has 20 observations and 160 variables

### Clean the data

Removing columns that contain NA missing values.

trainRaw <- trainRaw[, colSums(is.na(trainRaw)) == 0]   
dim(trainRaw)

## [1] 19622 93

names(trainRaw)

## [1] "X" "user\_name"   
## [3] "raw\_timestamp\_part\_1" "raw\_timestamp\_part\_2"   
## [5] "cvtd\_timestamp" "new\_window"   
## [7] "num\_window" "roll\_belt"   
## [9] "pitch\_belt" "yaw\_belt"   
## [11] "total\_accel\_belt" "kurtosis\_roll\_belt"   
## [13] "kurtosis\_picth\_belt" "kurtosis\_yaw\_belt"   
## [15] "skewness\_roll\_belt" "skewness\_roll\_belt.1"   
## [17] "skewness\_yaw\_belt" "max\_yaw\_belt"   
## [19] "min\_yaw\_belt" "amplitude\_yaw\_belt"   
## [21] "gyros\_belt\_x" "gyros\_belt\_y"   
## [23] "gyros\_belt\_z" "accel\_belt\_x"   
## [25] "accel\_belt\_y" "accel\_belt\_z"   
## [27] "magnet\_belt\_x" "magnet\_belt\_y"   
## [29] "magnet\_belt\_z" "roll\_arm"   
## [31] "pitch\_arm" "yaw\_arm"   
## [33] "total\_accel\_arm" "gyros\_arm\_x"   
## [35] "gyros\_arm\_y" "gyros\_arm\_z"   
## [37] "accel\_arm\_x" "accel\_arm\_y"   
## [39] "accel\_arm\_z" "magnet\_arm\_x"   
## [41] "magnet\_arm\_y" "magnet\_arm\_z"   
## [43] "kurtosis\_roll\_arm" "kurtosis\_picth\_arm"   
## [45] "kurtosis\_yaw\_arm" "skewness\_roll\_arm"   
## [47] "skewness\_pitch\_arm" "skewness\_yaw\_arm"   
## [49] "roll\_dumbbell" "pitch\_dumbbell"   
## [51] "yaw\_dumbbell" "kurtosis\_roll\_dumbbell"   
## [53] "kurtosis\_picth\_dumbbell" "kurtosis\_yaw\_dumbbell"   
## [55] "skewness\_roll\_dumbbell" "skewness\_pitch\_dumbbell"  
## [57] "skewness\_yaw\_dumbbell" "max\_yaw\_dumbbell"   
## [59] "min\_yaw\_dumbbell" "amplitude\_yaw\_dumbbell"   
## [61] "total\_accel\_dumbbell" "gyros\_dumbbell\_x"   
## [63] "gyros\_dumbbell\_y" "gyros\_dumbbell\_z"   
## [65] "accel\_dumbbell\_x" "accel\_dumbbell\_y"   
## [67] "accel\_dumbbell\_z" "magnet\_dumbbell\_x"   
## [69] "magnet\_dumbbell\_y" "magnet\_dumbbell\_z"   
## [71] "roll\_forearm" "pitch\_forearm"   
## [73] "yaw\_forearm" "kurtosis\_roll\_forearm"   
## [75] "kurtosis\_picth\_forearm" "kurtosis\_yaw\_forearm"   
## [77] "skewness\_roll\_forearm" "skewness\_pitch\_forearm"   
## [79] "skewness\_yaw\_forearm" "max\_yaw\_forearm"   
## [81] "min\_yaw\_forearm" "amplitude\_yaw\_forearm"   
## [83] "total\_accel\_forearm" "gyros\_forearm\_x"   
## [85] "gyros\_forearm\_y" "gyros\_forearm\_z"   
## [87] "accel\_forearm\_x" "accel\_forearm\_y"   
## [89] "accel\_forearm\_z" "magnet\_forearm\_x"   
## [91] "magnet\_forearm\_y" "magnet\_forearm\_z"   
## [93] "classe"

testRaw <- testRaw[, colSums(is.na(testRaw)) == 0]   
dim(testRaw)

## [1] 20 60

names(testRaw)

## [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" "problem\_id"

Removing columns that do not contribute much to the accelerometer measurements.

head(trainRaw$classe)

## [1] A A A A A A  
## Levels: A B C D E

classe <- trainRaw$classe  
trainRemove <- grepl("^X|timestamp|window", names(trainRaw))  
trainRemove

## [1] TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE  
## [12] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [23] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [34] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [45] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [56] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [67] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [78] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [89] FALSE FALSE FALSE FALSE FALSE

trainRaw <- trainRaw[, !trainRemove]  
dim(trainRaw)

## [1] 19622 87

trainCleaned <- trainRaw[, sapply(trainRaw, is.numeric)]  
dim(trainCleaned)

## [1] 19622 52

trainCleaned$classe <- classe  
testRemove <- grepl("^X|timestamp|window", names(testRaw))  
testRaw <- testRaw[, !testRemove]  
testCleaned <- testRaw[, sapply(testRaw, is.numeric)]  
dim(testCleaned)

## [1] 20 53

The training data set has 19622 observations and 53 remaining variables The testing data set has 20 observations and remaining 53 variables

### Split the data

Splitting the cleaned training set into a pure training data set (70%) and a validation data set (30%). Validation data will be used to conduct cross validation in future steps.

set.seed(100) # For reproducibile purpose  
inTrain <- createDataPartition(trainCleaned$classe, p=0.70, list=FALSE)  
trainData <- trainCleaned[inTrain, ]  
dim(trainData)

## [1] 13737 53

testData <- trainCleaned[-inTrain, ]  
dim(testData)

## [1] 5885 53

## Data Modeling

Using Random Forest algorithm to fit a predictive model for activity recognition. Random Forest automatically selects important variables and is robust to correlated covariates & outliers in general. A **3-fold cross validation and ntree = 100** is chosen when applying the algorithm.

controlRf <- trainControl(method="cv", 3)  
modelRf <- train(classe ~ ., data=trainData, method="rf", trControl=controlRf, ntree=100)  
modelRf

## Random Forest   
##   
## 13737 samples  
## 52 predictor  
## 5 classes: 'A', 'B', 'C', 'D', 'E'   
##   
## No pre-processing  
## Resampling: Cross-Validated (3 fold)   
## Summary of sample sizes: 9158, 9159, 9157   
## Resampling results across tuning parameters:  
##   
## mtry Accuracy Kappa   
## 2 0.9866787 0.9831456  
## 27 0.9879160 0.9847122  
## 52 0.9788169 0.9731963  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was mtry = 27.

Estimation performance of the model on the validation data set.

predictRf <- predict(modelRf, testData)  
confusionMatrix(testData$classe, predictRf)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1674 0 0 0 0  
## B 7 1131 1 0 0  
## C 0 4 1021 1 0  
## D 0 2 15 947 0  
## E 0 0 2 1 1079  
##   
## Overall Statistics  
##   
## Accuracy : 0.9944   
## 95% CI : (0.9921, 0.9961)  
## No Information Rate : 0.2856   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9929   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9958 0.9947 0.9827 0.9979 1.0000  
## Specificity 1.0000 0.9983 0.9990 0.9966 0.9994  
## Pos Pred Value 1.0000 0.9930 0.9951 0.9824 0.9972  
## Neg Pred Value 0.9983 0.9987 0.9963 0.9996 1.0000  
## Prevalence 0.2856 0.1932 0.1766 0.1613 0.1833  
## Detection Rate 0.2845 0.1922 0.1735 0.1609 0.1833  
## Detection Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Balanced Accuracy 0.9979 0.9965 0.9908 0.9972 0.9997

accuracy <- data.frame(postResample(predictRf, testData$classe))  
accuracy

## postResample.predictRf..testData.classe.  
## Accuracy 0.9943925  
## Kappa 0.9929060

oose <- 1 - as.numeric(confusionMatrix(testData$classe, predictRf)$overall[1])  
oose

## [1] 0.005607477

Estimated accuracy of the model is 99.44% and the estimated out-of-sample error is 0.56%.

## Predicting for Test Data Set

Applying the model to the original testing data set downloaded from the data source. We remove the problem\_id column first.

result <- predict(modelRf, testCleaned[, -length(names(testCleaned))])  
result

## [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

## Appendix: Figures

Decision Tree Visualization

treeModel <- rpart(classe ~ ., data=trainData, method="class")  
prp(treeModel)

