PMLproject

pd

Saturday, August 22, 2015

Abstract

The objective of this project is to build a machine learning model and use it to recognize human activities based on the Human Activity dataset from Groupware. Additional information for the data is available from the website at: http://groupware.les.inf.puc-rio.br/har. For this assignment, the provided data is used to build a model, train, and test the recognition of activity type that an individual performs. The Random forest method is used and the test results demonstartes a very high acuracy of the model's performance. The correct answers for the 20 test-case scenarios of this assignment prove the validity of the approach.

Data input and preprocessing

Import Data Sets - specifying the data sources and training / testing files destination. First, load the datasets from the links provided for training and test data. Preview the datasets. Some values containe a "#DIV/0!", " ", or "NA" value. Columns with mostly missing values do not contribute to the training/prediction. It is reasonable to sellect features that have complete columns of data. Windows and timestamp are irrelevant to this assignment. Privacy dictate to remove the user name. Changing the values to numeric simplifies processing.

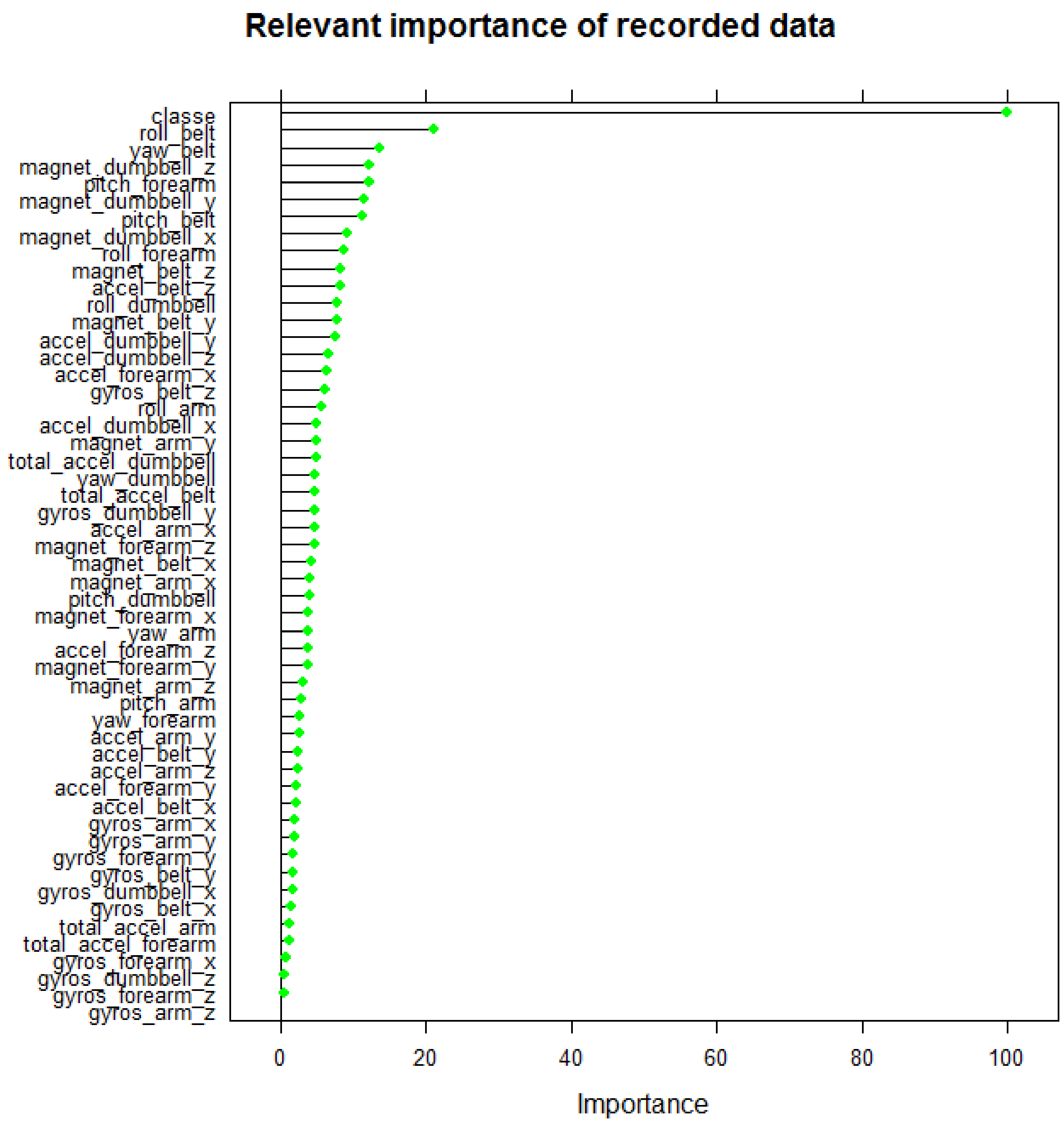
```
setwd("C:/Users/Plamen/Documents")
training_data <- read.csv('pml-training.csv', na.strings = c("#DIV/0!", "NA",
""))
evaluation_data <- read.csv('pml-test.csv', na.strings = c("#DIV/0!", "NA",
""))
for(i in c(8:ncol(training_data)-1))
  training_data[,i] = as.numeric(as.character(training_data[,i]))
for(i in c(8:ncol(evaluation_data)-1))
  evaluation_data[,i] = as.numeric(as.character(evaluation_data[,i]))
```

Examine the features in the dataset, determine and display the features. Allocate the data for training (75%) and testing (25%) subsets.

```
feature_set <- colnames(training_data[colSums(is.na(training_data)) == 0])[-
(1:7)]
model_data <- training_data[feature_set]
feature_set

##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"     "classe"
```

```
index <- createDataPartition(y=model_data$classe, p=0.75, list=FALSE )
training <- model_data[index,]
testing <- model_data[-index,]
suppressWarnings(suppressMessages(library(randomForest)))
random.forest <- train(training[,-57], training$classe,
tuneGrid=data.frame(mtry=3),
                        trControl=trainControl(method="none"))
plot(varImp(random.forest), col="green", main="Relevant importance of
recorded data")
```



Model training and testing

Build multiple (for example 10) random forests with 100 trees each. To speed up the processing, take advantage of R's parallel processing to build this model (multiple examples are available on www).

```
suppressWarnings(suppressMessages(library(doParallel)))
registerDoParallel()
suppressWarnings(suppressMessages(library(foreach)))
```

```
x <- training[-ncol(training)]
y <- training$classe
rf <- foreach(ntree=rep(100, 2), .combine=randomForest::combine,
.packages='randomForest') %dopar%
randomForest(x, y, ntree=ntree)
```

Examine the accuracy reports for training and test data to evaluate the overall performance of the model.

```
predict_1 <- predict(rf, newdata=training)
CM1 <- confusionMatrix(predict_1,training$classe)
CM1$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      1.0000000      1.0000000      0.9997494      1.0000000      0.2843457
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

```
predict_2 <- predict(rf, newdata=testing)
CM2 <- confusionMatrix(predict_2,testing$classe)
CM2$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##      0.9949021      0.9935517      0.9924837      0.9966983      0.2844617
## AccuracyPValue  McnemarPValue
##      0.0000000            NaN
```

Model performance and conclusions

The random forest algorithm appears to perform very well for predicting human activities from recorded accelerometers measurements during the activities. As can be seen from the confusion matrix, this model is very accurate. The correct answers for the 20 test-case scenarios of this assignment prove the validity of the approach.

Generate submission files

Prepare the output files for submission using Coursera provided code.

```
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")

write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}



x <- evaluation_data
x <- x[feature_set[feature_set!='classe']]
```

```r
answers <- predict(rf, newdata=x)

answers
```

```
##  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
```

```r
pml_write_files(answers)
```