Practical Machine Learning Project

E121977

Thursday, January 28, 2016

Data Preprocessing

Assignment Introduction

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement - a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Data Preprocessing

```
library(caret)
```

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

```
library(corrplot)
library(e1071)
library(randomForest)
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(rpart)
library(rpart.plot)
```

Download the Data

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

Read the Data

After downloading the data from the data source, we can read the raw training and test files into two data frames.

```
trainDataSet <- read.csv(sprintf("%s/data/pml-training.csv",getwd()))
testDataSet <- read.csv(sprintf("%s/data/pml-testing.csv",getwd()))
dim(trainDataSet)
```

```
## [1] 19622   160
```

```
dim(testDataSet)
```

```
## [1]  20 160
```

The training data set contains 19,622 observations and 160 variables. The testing data set contains 20 observations and 160 variables. The "classe" variable in the training set is the outcome to predict.

Clean the data

Clean and tidy the data by removing observations with missing values and removing meaningless variables.

```
sum(complete.cases(trainDataSet))
```

```
## [1] 406
```

Remove columns that contain NA missing values.

```
trainDataSet <- trainDataSet[, colSums(is.na(trainDataSet)) == 0]
testDataSet <- testDataSet[, colSums(is.na(testDataSet)) == 0]
```

Remove columns that do not pertain to the analysis, remove clutter or noise.

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

Now, the cleaned training data set contains 19,622 observations and 53 variables. The testing data set contains 20 observations and 53 variables. The "classe" variable is still in the cleaned training set.

Split the data into training and testing sets

Split the cleaned training set into a training data set (70%) and a validation/testing data set (30%). We will use the validation/testing data set to conduct cross validation in future steps.

```
set.seed(22519) # For reproducibile purpose
inTrain <- createDataPartition(trainCleaned$classe, p=0.70, list=F)
trainData <- trainCleaned[inTrain, ]
testData <- trainCleaned[-inTrain, ]
```

Data Modeling

Fit a predictive model for activity recognition using the **Random Forest** algorithm. This step automatically selects important variables and is robust to correlated covariates & outliers in general. We will use **5-fold cross validation** when applying the algorithm.

```
controlRf <- trainControl(method="cv", 5)
modelRf <- train(classe ~ ., data=trainData, method="rf", trControl=controlRf, ntree=250)
modelRf
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10989, 10989, 10991, 10990, 10989
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9908278  0.9883961  0.001675093  0.002121380
##   27    0.9911190  0.9887646  0.001755971  0.002222771
##   52    0.9840572  0.9798290  0.003497420  0.004425063
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

Estimate the model performance 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 1672    1    0    0    1
##          B    5 1129    5    0    0
##          C    0    1 1020    5    0
##          D    0    0   13  949    2
##          E    0    0    1    6 1075
##
## Overall Statistics
##
##                Accuracy : 0.9932
##                  95% CI : (0.9908, 0.9951)
##     No Information Rate : 0.285
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9914
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9970   0.9982   0.9817   0.9885   0.9972
## Specificity            0.9995   0.9979   0.9988   0.9970   0.9985
## Pos Pred Value         0.9988   0.9912   0.9942   0.9844   0.9935
## Neg Pred Value         0.9988   0.9996   0.9961   0.9978   0.9994
## Prevalence             0.2850   0.1922   0.1766   0.1631   0.1832
## Detection Rate         0.2841   0.1918   0.1733   0.1613   0.1827
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9983   0.9981   0.9902   0.9927   0.9979
```

```
accuracy <- postResample(predictRf, testData$classe)
accuracy
```

```
##  Accuracy     Kappa
## 0.9932031 0.9914024
```

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

```
## [1] 0.006796941
```

We can see estimated accuracy of the model is 99.30% and the estimated out-of-sample error is 0.70%.

Predicting for Test Data Set

Apply the model to the original testing data set downloaded from the data source and remove the `problem_id` column

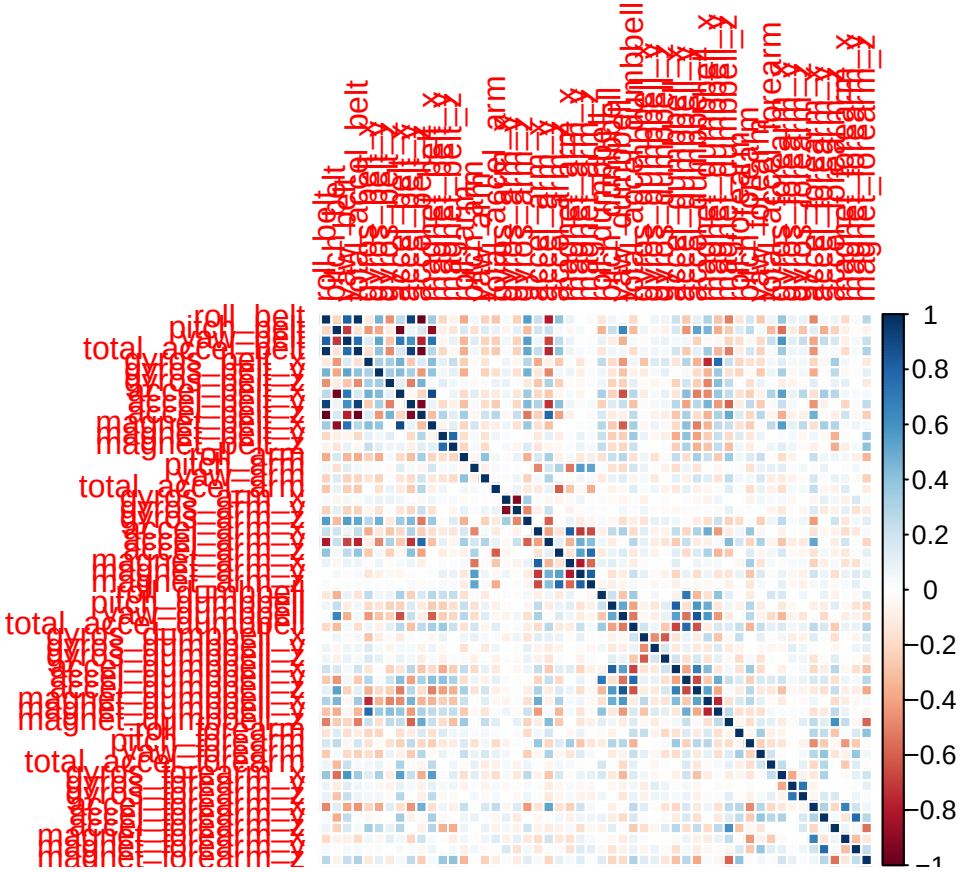
```
result <- predict(modelRf, testCleaned[, -length(names(testCleaned))])
result
```

```
##  [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

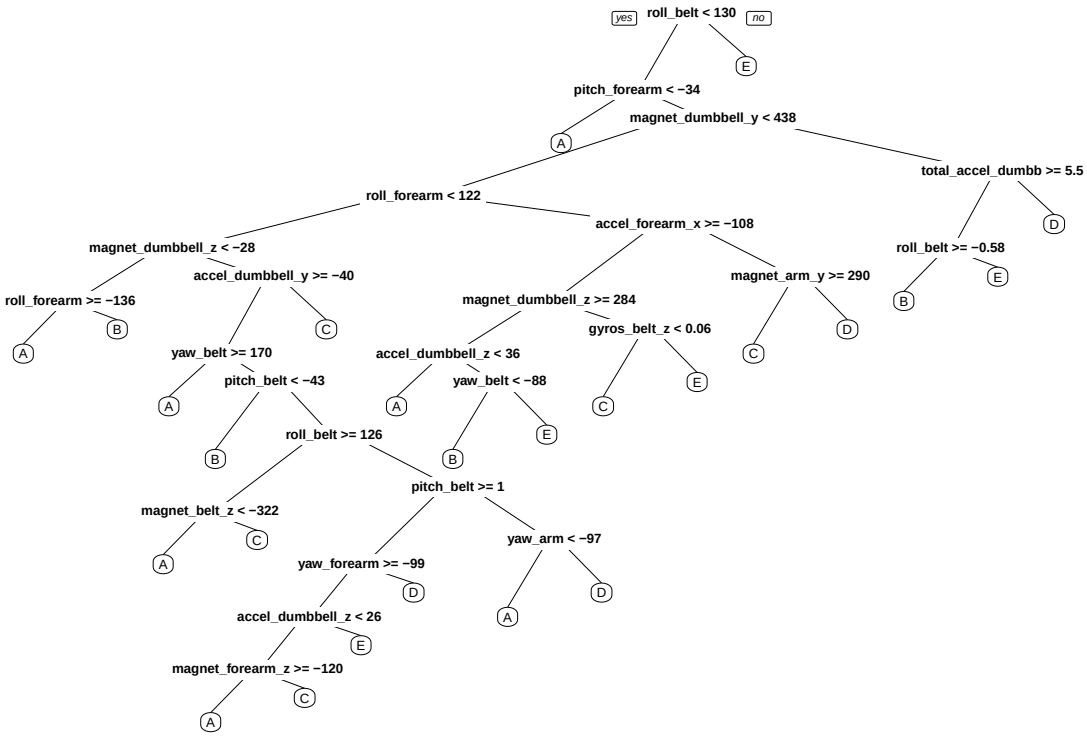
1. Correlation Matrix Visualization

```
corrPlot <- cor(trainData[, -length(names(trainData))])
corrplot(corrPlot, method="color")
```



2. Decision Tree Visualization

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


yes
roll_belt < 130
no
E
pitch_forearm < −34
magnet_dumbbell_y < 438
A
total_accel_dumbb >= 5.5
roll_forearm < 122
accel_forearm_x >= −108
D
magnet_dumbbell_z < −28
roll_belt >= −0.58
accel_dumbbell_y >= −40
magnet_arm_y >= 290
E
roll_forearm >= −136
magnet_dumbbell_z >= 284
B
gyros_belt_z < 0.06
A
B
C
yaw_belt >= 170
accel_dumbbell_z < 36
C
D
pitch_belt < −43
yaw_belt < −88
A
A
E
C
E
roll_belt >= 126
B
B
pitch_belt >= 1
magnet_belt_z < −322
yaw_arm < −97
A
C
yaw_forearm >= −99
A
D
D
accel_dumbbell_z < 26
E
magnet_forearm_z >= −120
C
A