Practical Machine Learning Course Project

AK

April 29, 2018

Prediction of the manner in which 6 participants did the Dumbbell Biceps Curl

Assignment

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, the goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to develop a model to predict the manner in which they did the Dumbbell Biceps Curl. Are they doing it correctly according to the specifications ?

1. Class A: According to the specifications
2. Class B: throwing the elbows to the front

3. Class C: lifting the dumbbell only halfway

4. Class D: lowering the dumbbell only halfway

5. Class E: throwing the hips to the front

Synopsis

The objective of this exercise is to develop a model to predict the manner in which 6 participants did the Dumbbell Biceps Curl. Are they doing it in according to the specifications ?

The dataset which I use was extracted from http://groupware.les.inf.puc-rio.br/har. The training file contains 19622 observations and 160 varialbles. To increase the accruacy of the forecast model, preprocessing is required to exclude all columns containing (01) 60% or more NA and (02) Near Zero variance. Furthermore subsetting the training data set to 2 sets: Training and Cross-Validation to assess the accuracy of the models.

Conclusion - Selection of Model:

I am testing 2 methods: RandomForest vs Rpart (Recursive partitioning for classification)

My analysis shows that RandomForest Model has a much higher accuracy than Rpart model in both Training and Out of Sample (cross validation) predictions.

Training Set Accuracy Rates:

RandomForest: 1 Rpart: 0.4979298

Cross Validation Set Accuracy Rates:

RandomForest: 0.9964313 Rpart: 0.4863625

For RandomForest Model, the out of sample (cross validation) accuracy rate is very close to the training rate. This shows that the model behaves consistently.

RandomForest Training Accuracy Rate: 1 RandomForest Out of Sample (Cross Validation) Accuracy Rate: 0.9964313. The error rate is 0.004

Results for Test Quiz

The forecast of the test set match the results in the quiz section.

print(predict(modrf,newdata=testing))

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

Loading and Processing of Data

Training Data Source: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

Testing Data Source: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Training Data File has 19622 observations and 160 varialbles

Loading of required packages and the datasets

```
# Loading required packages

library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.4

## Loading required package: lattice

## Warning: package 'lattice' was built under R version 3.4.4

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.4.4
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.4

## 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
```

```
# Loading data sets
temp<-tempfile()
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", temp)
train<-read.csv(temp)
temp<-tempfile()
```

```
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", temp)
testing<-read.csv(temp)

# Subset the train dataset to 2 subsets: Train and Validation

set.seed(34567)
intrain<-createDataPartition(train$classe,p=0.8,list=FALSE)
train1<-train[intrain,]
validation<-train[-intrain,]

dim(train)
```

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

```
dim(train1)
```

```
## [1] 15699   160
```

```
dim(validation)
```

```
## [1] 3923  160
```

Pre-processing - Exclude all columns containing (01) 60% or more NA/“” and (02) Near Zero variance. Furthermore exclude certain columns that do not appear to contribute the model.

```
# Near Zero Variance COlumns

nearzero<-nearZeroVar(train1)
train1<-train1[,-nearzero]

# Columns containing 60% or more NA or ""

training<- train1[,!(colSums(is.na(train1)|train1=="")/nrow(train1)>.6)]

# Exclude columns that do not appear to contribute to the model.

training<-training[,!names(training) %in% c("X", "user_name", "raw_timestamp_part_1", "raw_timestamp_pa
```

Model Selection

Modeling

```
# RandomForest

modrf<-randomForest(classe~.,data=training,importance=TRUE,prox=FALSE,ntrees=10)

# rpart

modrpart<-train(classe~.,data=training, method="rpart")
```

Selection

```
# randomforest

confusionMatrix(predict(modrf,newdata=training),training$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4464    0    0    0    0
##          B    0 3038    0    0    0
##          C    0    0 2738    0    0
##          D    0    0    0 2573    0
##          E    0    0    0    0 2886
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9998, 1)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```
confusionMatrix(predict(modrf,newdata=validation),validation$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    1    0    0    0
##          B    0  755    3    0    0
##          C    0    3  680    4    2
##          D    0    0    1  639    0
##          E    0    0    0    0  719
##
## Overall Statistics
##
##                Accuracy : 0.9964
##                  95% CI : (0.994, 0.998)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9955
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9942   0.9938   0.9972
## Specificity            0.9996   0.9991   0.9972   0.9997   1.0000
## Pos Pred Value         0.9991   0.9960   0.9869   0.9984   1.0000
## Neg Pred Value         1.0000   0.9987   0.9988   0.9988   0.9994
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1925   0.1733   0.1629   0.1833
## Detection Prevalence   0.2847   0.1932   0.1756   0.1631   0.1833
## Balanced Accuracy      0.9998   0.9969   0.9957   0.9967   0.9986
```

```r
confusionMatrix(predict(modrf,newdata=training),training$classe)$overall["Accuracy"]
```

```
## Accuracy
##        1
```

```r
confusionMatrix(predict(modrf,newdata=validation),validation$classe)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9964313
```

```r
# rpart

confusionMatrix(predict(modrpart,newdata=training),training$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4062 1234 1257 1120  414
##          B   70 1043   87  475  389
##          C  321  761 1394  978  765
##          D    0    0    0    0    0
##          E   11    0    0    0 1318
##
## Overall Statistics
##
##                Accuracy : 0.4979
##                  95% CI : (0.4901, 0.5058)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3443
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9099  0.34332   0.5091   0.0000  0.45669
## Specificity            0.6417  0.91936   0.7820   1.0000  0.99914
## Pos Pred Value         0.5023  0.50533   0.3304      NaN  0.99172
## Neg Pred Value         0.9472  0.85369   0.8829   0.8361  0.89088
## Prevalence             0.2843  0.19352   0.1744   0.1639  0.18383
## Detection Rate         0.2587  0.06644   0.0888   0.0000  0.08395
## Detection Prevalence   0.5151  0.13147   0.2687   0.0000  0.08466
## Balanced Accuracy      0.7758  0.63134   0.6456   0.5000  0.72791
```

```
confusionMatrix(predict(modrpart,newdata=validation),validation$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1002  319  328  308  107
##          B   25  259   22  105  106
##          C   86  181  334  230  195
##          D    0    0    0    0    0
##          E    3    0    0    0  313
##
## Overall Statistics
##
##                Accuracy : 0.4864
##                  95% CI : (0.4706, 0.5021)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3281
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8978  0.34124  0.48830   0.0000  0.43412
## Specificity            0.6217  0.91846  0.78635   1.0000  0.99906
## Pos Pred Value         0.4855  0.50097  0.32554      NaN  0.99051
## Neg Pred Value         0.9387  0.85320  0.87919   0.8361  0.88689
## Prevalence             0.2845  0.19347  0.17436   0.1639  0.18379
## Detection Rate         0.2554  0.06602  0.08514   0.0000  0.07979
## Detection Prevalence   0.5261  0.13179  0.26153   0.0000  0.08055
## Balanced Accuracy      0.7598  0.62985  0.63733   0.5000  0.71659
```

```
confusionMatrix(predict(modrpart,newdata=training),training$classe)$overall["Accuracy"]
```

```
##  Accuracy
## 0.4979298
```

```
confusionMatrix(predict(modrpart,newdata=validation),validation$classe)$overall["Accuracy"]
```

```
##  Accuracy
## 0.4863625
```

Prediction for the test quiz

```
print(predict(modrf,newdata=testing))
```

```
##  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
```