Coursera Machine Learning Project Writeup

Ven Reddy

Thursday, September 24, 2015

Assignment

Fitness devices such as Jawbone Up, Nike Fuelband, and Fitbit provide a means for enthusiasts to improve their health, find patternsin their behavior. People use these devices to quantify how much of an given activity they do, but not how well they do that particular activity. Our goals is to build a predictor that can tells us how well a person has done a particular exercise.

Data

The dataset was obtained from:

Human Activity Recognition website @ http://groupware.les.inf.puc-rio.br/har

The training data for this project is available here: http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data is available here: http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Processing

We need to clean up the data by removing columns that are empty, contain mostly NA, and have little to no impact on the training algorithm.

```
trainingRaw <- read.csv("pml-training.csv")
testingRaw <- read.csv("pml-testing.csv")

trainingRaw <- data.frame(trainingRaw)
trainingRaw <- trainingRaw[, !is.na(colSums(trainingRaw != 0)) & colSums(trainingRaw != 0) > 0]

trainRaw <- trainingRaw %>% select(-starts_with("kurtosis"),-starts_with("skewness"), -starts_with("max
                                   -starts_with("min_yaw"), -starts_with("amplitude_yaw"), -starts_with
                                   -contains("timestamp"), -contains("window"))
```

Modeling using Random Forest and Cross Validation

We now split the training data into two parts. 75% for training and 25% for testing the model. The Random Forest method with 3 fold Cross Validation is used. I had used the 5 fold Cross Validation but the amount of time it took for the improvement in accuracy didn't seem justified; an improvement of approximately of 0.001.

```
# Set seed so we results are reproducible
set.seed(1234)
train <- createDataPartition(trainRaw$classe, p=0.75, list=FALSE)
trainData <- trainRaw[train,]
testData <- trainRaw[-train,]
```

```r
# remove unwanted variables
trc = trainControl( method="cv", number=3, allowParallel=TRUE, verboseIter=TRUE)

# create the model
mtrain <- train(classe ~., trainData, method="rf", trControl=trc)
```

```
## + Fold1: mtry= 2
## - Fold1: mtry= 2
## + Fold1: mtry=27
## - Fold1: mtry=27
## + Fold1: mtry=52
## - Fold1: mtry=52
## + Fold2: mtry= 2
## - Fold2: mtry= 2
## + Fold2: mtry=27
## - Fold2: mtry=27
## + Fold2: mtry=52
## - Fold2: mtry=52
## + Fold3: mtry= 2
## - Fold3: mtry= 2
## + Fold3: mtry=27
## - Fold3: mtry=27
## + Fold3: mtry=52
## - Fold3: mtry=52
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 27 on full training set
```

```r
# get predictions and create Confusion matrix to see how well classifier performance is
pred <- predict(mtrain, testData)
cm <- confusionMatrix(pred, testData$classe)
```

Results

```r
print(mtrain)
```

```
## Random Forest
##
## 14718 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (3 fold)
## Summary of sample sizes: 9811, 9812, 9813
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9889253  0.9859891  0.001892565  0.002394077
##   27    0.9898085  0.9871065  0.001943737  0.002460195
```

```
##   52    0.9848488  0.9808305  0.004177884  0.005288806
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

```
print(cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    7    0    0    0
##          B    0  939    9    1    0
##          C    0    3  844    8    2
##          D    0    0    2  795    0
##          E    0    0    0    0  899
##
## Overall Statistics
##
##                Accuracy : 0.9935
##                  95% CI : (0.9908, 0.9955)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9917
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9895   0.9871   0.9888   0.9978
## Specificity            0.9980   0.9975   0.9968   0.9995   1.0000
## Pos Pred Value         0.9950   0.9895   0.9848   0.9975   1.0000
## Neg Pred Value         1.0000   0.9975   0.9973   0.9978   0.9995
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1915   0.1721   0.1621   0.1833
## Detection Prevalence   0.2859   0.1935   0.1748   0.1625   0.1833
## Balanced Accuracy      0.9990   0.9935   0.9920   0.9942   0.9989
```

Confusion Matrix and Statistics

Reference

Prediction A B C D E

A 1395 7 0 0 0

B 0 939 9 1 0

C 0 3 844 8 2

D 0 0 2 795 0

E 0 0 0 0 899

Overall Statistics

Accuracy : 0.9935

95% CI : (0.9908, 0.9955)

No Information Rate : 0.2845

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.9917

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: A Class: B Class: C Class: D Class: E

Sensitivity 1.0000 0.9895 0.9871 0.9888 0.9978

Specificity 0.9980 0.9975 0.9968 0.9995 1.0000

Pos Pred Value 0.9950 0.9895 0.9848 0.9975 1.0000

Neg Pred Value 1.0000 0.9975 0.9973 0.9978 0.9995

Prevalence 0.2845 0.1935 0.1743 0.1639 0.1837

Detection Rate 0.2845 0.1915 0.1721 0.1621 0.1833

Predict the test cases

```
testingRaw <- data.frame(testingRaw)
testingRaw <- testingRaw[, !is.na(colSums(testingRaw != 0)) & colSums(testingRaw != 0) > 0]

# Remove unwanted columns
testRaw <- testingRaw %>% select(-starts_with("X"), -contains("timestamp"), -contains("window"), -conta

#Predict the classe outcome using our model
pred2 <- predict(mtrain, testRaw)
print(pred2)
```

```
##  [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
```

Conclusion

The answers were sumbitted to Coursera and our model predicted all 20 cases 100% correctly.

```
# This code is used to create the 20 .txt files for project submission
if (0) {
answers <- as.character(pred2);
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)
        }
}
pml_write_files(answers);
}
```