Practical Machine Learning Project

by: Jerich King

I.Summary

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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

II.Data Pre-processing

Loading the required files, and library.

```
library('caret')
library('randomForest')
train <- read.csv(url("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-trainin
g.csv"),na.strings=c("NA","#DIV/0!", ""))
validation <- read.csv(url("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-tes
ting.csv"),na.strings=c("NA","#DIV/0!", ""))
```

III.Data Cleaning

Removing unnecessary data when doing prediction.

```
#Removing Columns
train <- train[,-c(1:7)]

#Removing NA's
train <- train[, colSums(is.na(train)) == 0]

#Removing near zero covariates
nzv <- nearZeroVar(train,saveMetrics=TRUE)
train <- train[, nzv$nzv==FALSE]
```

IV.Data Partitioning

```
set.seed(1)
partition <- createDataPartition(train$classe, p=0.8, list=FALSE)
training <- train[partition, ]
testing <- train[-partition, ]
```

V.Random Forest Model

```
set.seed(1)
traincontrolset <- trainControl(method="cv", number=3, verboseIter=FALSE)
RFM <- train(classe ~ ., data=training, method="rf", trControl=traincontrolset)
RFM
```

```
## Random Forest
##
## 15699 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (3 fold)
## Summary of sample sizes: 10466, 10466, 10466
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9916555  0.9894442
##   27    0.9900631  0.9874298
##   52    0.9833110  0.9788872
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
#Predicting
RFM_predict <- predict(RFM, testing)
confusionMatrix(testing$classe, RFM_predict)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1115    1    0    0    0
##          B    4  754    1    0    0
##          C    0    4  680    0    0
##          D    0    0    9  634    0
##          E    0    0    0    0  721
##
## Overall Statistics
##
##                Accuracy : 0.9952
##                  95% CI : (0.9924, 0.9971)
##     No Information Rate : 0.2852
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9939
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9964   0.9934   0.9855   1.0000   1.0000
## Specificity            0.9996   0.9984   0.9988   0.9973   1.0000
## Pos Pred Value         0.9991   0.9934   0.9942   0.9860   1.0000
## Neg Pred Value         0.9986   0.9984   0.9969   1.0000   1.0000
## Prevalence             0.2852   0.1935   0.1759   0.1616   0.1838
## Detection Rate         0.2842   0.1922   0.1733   0.1616   0.1838
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9980   0.9959   0.9921   0.9986   1.0000
```

VI.Results

RFM produced a result of 99.52% accuracy and a Kappa of 0.9939.

VII.Model Validation

```
results <- predict(RFM, newdata=validation)
results
```

```
##  [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
```