Practical Machine Learning Course Project

Weiying

8/25/2020

Introduction and 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. Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E). Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes.

Data, EDA and Clean

Load the csv training and testing data respective. By looking at the data, decided to get rid of the identity, user name, time stamp and windows columns. Further, since we have no logic on how to compensate the data with NAs, simply exclude them at the moment, after we did the modeling, we can decide if we need them or not.

```r
# training data
PML_training <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"),
                         na.strings=c("NA","#DIV/0!",""))
PML_testing <- read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"),
                        na.strings=c("NA","#DIV/0!",""))

# by looking at the data, get rid of timestamp data, identity related columns for this modeling
PML_training <- PML_training[,8:length(PML_training)]
PML_testing <- PML_testing[,8:length(PML_testing)]

# exclude columns with NAs
PML_training <- PML_training[, colSums(is.na(PML_training)) == 0]
PML_testing <- PML_testing[, colSums(is.na(PML_testing)) == 0]

table(PML_training$classe)
```

```
##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

Prepare Data for Training

Separate the training data into a training set (70% of training data) and a validation set (30% of training data).

```
set.seed(1234)
trainset <- createDataPartition(PML_training$classe, p = 0.7, list = FALSE)
Training <- PML_training[trainset, ]
Validation <- PML_training[-trainset, ]
```

Build a Random Forest Model

Using a random forest model with the training data to predict the classe.

```
# training with random forest model
rfModel <- randomForest(classe ~., data=Training, type="class")

rfModel
```

```
## 
## Call:
##  randomForest(formula = classe ~ ., data = Training, type = "class") 
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
## 
##         OOB estimate of  error rate: 0.49%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    2    0    0    0 0.0005120328
## B   10 2644    4    0    0 0.0052671181
## C    0   15 2378    3    0 0.0075125209
## D    1    0   24 2225    2 0.0119893428
## E    0    0    1    5 2519 0.0023762376
```

Use the seperated validation data to validate the model.

```
# validate the model with the validation data
validate <- predict(rfModel, newdata = Validation, type="class")
confusionMatrix(validate, Validation$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    3    0    0    0
##          B    0 1132   14    0    0
##          C    0    4 1012    7    0
##          D    0    0    0  956    0
##          E    0    0    0    1 1082
## 
## Overall Statistics
```

```
## 
##                Accuracy : 0.9951
##                  95% CI : (0.9929, 0.9967)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9938
## 
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9939   0.9864   0.9917   1.0000
## Specificity            0.9993   0.9971   0.9977   1.0000   0.9998
## Pos Pred Value         0.9982   0.9878   0.9892   1.0000   0.9991
## Neg Pred Value         1.0000   0.9985   0.9971   0.9984   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1924   0.1720   0.1624   0.1839
## Detection Prevalence   0.2850   0.1947   0.1738   0.1624   0.1840
## Balanced Accuracy      0.9996   0.9955   0.9920   0.9959   0.9999
```

The model has an accuray of 0.9936.

Prediction exercise

Perform the prediction:

```r
pred <- predict(rfModel, newdata = PML_testing, type="class")
```

The final outcome show below.

```r
pred
```

```
##  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
```