coursera_ml_project

Abhishek Dubey

4/8/2018

```
## Warning: package 'caret' was built under R version 3.4.3
```

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

```
## Warning in as.POSIXlt.POSIXct(Sys.time()): unknown timezone 'zone/tz/2018c.
## 1.0/zoneinfo/Asia/Kolkata'
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

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. In this project, goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. The goal of project is to predict the manner in which participants did the exercise. This is the “classe” variable in the training set. We will be using different classification models and will use model with best accuracy to predict the values on test dataset.

Data Pre-Processing

```
setwd("/Users/abhishekdubey/Desktop/Learning/Coursera/Practical_Machine_Learning/co
ursera_ml_project")
training <- read.csv("pml-training.csv")
testing <- read.csv("pml-testing.csv")

# Partition the training dataset into "Training" & "Validation"
inTrain  <- createDataPartition(training$classe, p=0.7, list=FALSE)
trainingData <- training[inTrain, ]
validationData  <- training[-inTrain, ]
dim(trainingData)
```

```
## [1] 13737   160
```

```
dim(validationData)
```

```
## [1] 5885  160
```

Data Cleaning- Remove records with Near Zero Variance & records with NAs, Remove Identification records

```
#Remove Near Zero Variance Columns
near.zero.var <- nearZeroVar(training)
trainingData <- trainingData[, -near.zero.var]
validationData <- validationData[, -near.zero.var]
dim(trainingData)
```

```
## [1] 13737   100
```

```
dim(validationData)
```

```
## [1] 5885  100
```

```
#Remove Columns that are largely NAs
all.na <- sapply(trainingData, function(x) mean(is.na(x))) > 0.95
trainingData <- trainingData[, all.na==FALSE]
validationData  <- validationData[, all.na==FALSE]
dim(trainingData)
```

```
## [1] 13737    59
```

```
dim(validationData)
```

```
## [1] 5885   59
```

```
#Remove identification only variables (columns 1 to 5)
trainingData <- trainingData[, -(1:5)]
validationData <- validationData[, -(1:5)]
dim(trainingData)
```

```
## [1] 13737    54
```

```
dim(validationData)
```

```
## [1] 5885   54
```

Prediction Model Building

Method: Generalized Boosted Model

model fit

```
set.seed(13434)
modFit.gbm <- train(classe ~ ., data=trainingData, method = "gbm",
                    trControl = trainControl(method = "repeatedcv", number = 5, rep
eats = 1),
                    verbose = FALSE)
modFit.gbm$finalModel
```

```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 53 predictors of which 40 had non-zero influence.
```

```
# Prediction on Validation dataset
predict.gbm <- predict(modFit.gbm, newdata=validationData)
confusionMatrix.gbm <- confusionMatrix(predict.gbm, validationData$classe)
confusionMatrix.gbm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1670    5    0    1    0
##          B    3 1121    7    4    2
##          C    1   12 1014    9    3
##          D    0    1    3  950   10
##          E    0    0    2    0 1067
##
## Overall Statistics
##
##                Accuracy : 0.9893
##                  95% CI : (0.9863, 0.9918)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9865
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9976   0.9842   0.9883   0.9855   0.9861
## Specificity            0.9986   0.9966   0.9949   0.9972   0.9996
## Pos Pred Value         0.9964   0.9859   0.9759   0.9855   0.9981
## Neg Pred Value         0.9990   0.9962   0.9975   0.9972   0.9969
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2838   0.1905   0.1723   0.1614   0.1813
## Detection Prevalence   0.2848   0.1932   0.1766   0.1638   0.1816
## Balanced Accuracy      0.9981   0.9904   0.9916   0.9913   0.9929
```

Method: Random Forest

```
set.seed(13434)
modFit.rf <- train(classe ~ ., data=trainingData, method="rf",
                   trControl=trainControl(method="cv", number=3, verboseIter=FALSE
))
modFit.rf$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.19%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3905    0    0    0    1 0.0002560164
## B    8 2647    2    1    0 0.0041384500
## C    0    4 2392    0    0 0.0016694491
## D    0    0    4 2247    1 0.0022202487
## E    0    0    0    5 2520 0.0019801980
```

```
# Prediction on Validation dataset
predict.rf <- predict(modFit.rf, newdata=validationData)
confusionMatrix.rf <- confusionMatrix(predict.rf, validationData$classe)
confusionMatrix.rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    1    0    0    0
##          B    1 1138    2    0    0
##          C    0    0 1024    2    0
##          D    0    0    0  962    1
##          E    0    0    0    0 1081
##
## Overall Statistics
##
##                Accuracy : 0.9988
##                  95% CI : (0.9976, 0.9995)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9985
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9991   0.9981   0.9979   0.9991
## Specificity            0.9998   0.9994   0.9996   0.9998   1.0000
## Pos Pred Value         0.9994   0.9974   0.9981   0.9990   1.0000
## Neg Pred Value         0.9998   0.9998   0.9996   0.9996   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1934   0.1740   0.1635   0.1837
## Detection Prevalence   0.2845   0.1939   0.1743   0.1636   0.1837
## Balanced Accuracy      0.9996   0.9992   0.9988   0.9989   0.9995
```

Apply Best Model to Test Dataset

The accuracy of Random Forest model is highest and is >99%. We will predict values on Test dataset from Random Forest Model.

```
predict.final <- predict(modFit.rf, newdata=testing)
predict.final
```

```
##  [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
```