Week 4 Practical Machine Learning Project

June Kieu

September 2, 2019

Summary

The goal of this project is to predict the manner in which users did the exercise. This is variable *classe* in training set. This report will describe my process of building the model.

Loading Data

The first step is to load the data. There are cells with values of "NA" and "#DIV/0!" or missing. I decided to replace them with the median of the variables.

```r
rm(list=ls())
setwd("C:/Users/June Kieu/Downloads")
getwd()
```

```
## [1] "C:/Users/June Kieu/Downloads"
```

```r
training <- read.csv("pml-training.csv",sep = ",",na.strings=c("NA","","#DIV/0!"))
training <- training[-c(1:8)]#taking out the first 8 columns as they are not real predictors
testing <- read.csv("pml-testing.csv",sep = ",",na.strings=c("NA","","#DIV/0!"))
testing <- testing[-c(1:8)]#taking out the first 8 columns as they are not real predictors
dim(training);dim(testing)
```

```
## [1] 19622   152
```

```
## [1]  20 152
```

```r
#Replacing all NA cells with median value of the corresponding
#column for both training and testing set
for(i in 1:151){
  training[is.na(training[,i]), i] <- median(training[,i], na.rm = TRUE)
}
for(i in 1:151){
  testing[is.na(testing[,i]), i] <- median(testing[,i], na.rm = TRUE)
}
```

Preparing data set for modeling

Training data set consists of 19,622 observations; using all of these to build the classification model is not recommended; as we cannot evaluate the model performance. Thus, I used *createDataPartition* function in package *caret* to split *training* data set into *TrainSet* and *TestSet*.

Also, there are variables with no or not significant variance; I excluded those out of *TrainSet* and *TestSet*. There are only 51 independent variables brought in the model.

```r
require(caret)
```

```
## Loading required package: caret
```

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

```
## Loading required package: ggplot2
```

```r
require(ggplot2)
inTrain <- createDataPartition(training$classe,p=.6,list = FALSE)
TrainSet <- training[inTrain,]
TestSet <- training[-inTrain,]
###eliminating variables with near zero variance
NZV <- nearZeroVar(TrainSet)
TrainSet <- TrainSet[, -NZV]
TestSet <- TestSet[, -NZV]
testing1  <- testing[, -NZV]
dim(TrainSet)
```

```
## [1] 11776    52
```

Modeling Step

Because the processing time for "rf" method using *caret* package is quite long, I decided to use "ranger" method instead. One of the parameters in *train* function regulates the method of cross validation, I decided to use *repeatedcv* method with 10-fold cross validations, which means dividing the data into 10 subsets, using 9 of them to train the model and 1 to test the performance. The process is repeated 3 times.

Model performance is determined on *TestSet*, looking at confusion matrix, we could see that this model predicts *classe* pretty precisely: 2231/2232 (there are 2232 *actual* obs with Classe A) are predicted correctly as Classe A; similarly, 1515/1524 are predicted correctly as Classe B, 1360/1368 are predicted correctly as classe C, only 5 observations out of 1286 *actual* Classe D observations are misclassified, and this number is 4/1442 observations for classe E. Model's overall accuracy is 99.73%.

```r
set.seed(123)
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 3)
RFMod <- train(classe~.,
                    data=TrainSet,
                    method="ranger",
                    trControl=fitControl,
               importance = "impurity")
```

```
## Growing trees.. Progress: 83%. Estimated remaining time: 6 seconds.
## Growing trees.. Progress: 88%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 88%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 88%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 86%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 86%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 88%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 82%. Estimated remaining time: 6 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 85%. Estimated remaining time: 5 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
```

```
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 86%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 88%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 86%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 82%. Estimated remaining time: 6 seconds.
## Growing trees.. Progress: 87%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 88%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 86%. Estimated remaining time: 4 seconds.
## Growing trees.. Progress: 89%. Estimated remaining time: 3 seconds.
## Growing trees.. Progress: 90%. Estimated remaining time: 3 seconds.
## Growing trees.. Progress: 90%. Estimated remaining time: 3 seconds.
```

```r
preRFMod <- predict(RFMod,newdata=TestSet)
confusionMatrix(preRFMod,TestSet$classe)
```

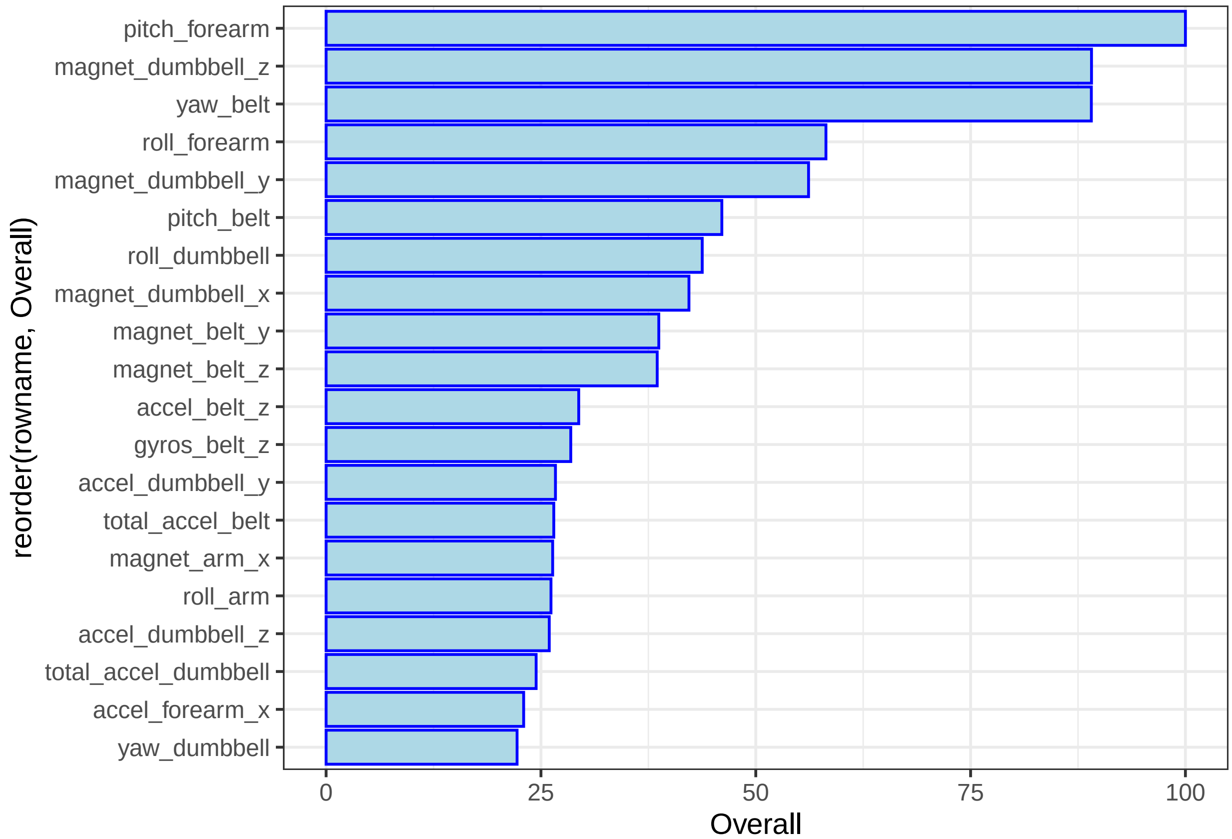
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2229    5    0    0    0
##          B    0 1509   19    0    1
##          C    0    4 1349   12    2
##          D    0    0    0 1274    9
##          E    3    0    0    0 1430
##
## Overall Statistics
##
##                Accuracy : 0.993
##                  95% CI : (0.9909, 0.9947)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9911
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9987   0.9941   0.9861   0.9907   0.9917
## Specificity            0.9991   0.9968   0.9972   0.9986   0.9995
## Pos Pred Value         0.9978   0.9869   0.9868   0.9930   0.9979
## Neg Pred Value         0.9995   0.9986   0.9971   0.9982   0.9981
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2841   0.1923   0.1719   0.1624   0.1823
## Detection Prevalence   0.2847   0.1949   0.1742   0.1635   0.1826
## Balanced Accuracy      0.9989   0.9955   0.9917   0.9946   0.9956
```

Important Variables

pitch_forearm, yaw_belt, magnet_dumbbell_z, roll_forearm and roll_forearm are among top variables of RFMod. Below is a visualization of my model's top 20 variables and their importances. The model then is used for *testing* set to predict classe for its 20 observations.

```
## Loading required package: e1071
```

```
## Loading required package: dplyr
##
## 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
## Loading required package: tidyverse
## -- Attaching packages ---------------------------------------------------------------- tidyverse 1.2.
## v tibble  2.1.3     v purrr   0.3.2
## v tidyr   0.8.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
## -- Conflicts ------------------------------------------------------------------- tidyverse_conflicts(
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
## ranger variable importance
##
##   only 20 most important variables shown (out of 51)
##
##                      Overall
## pitch_forearm         100.00
## magnet_dumbbell_z      89.07
## yaw_belt               89.05
## roll_forearm           58.17
## magnet_dumbbell_y      56.14
## pitch_belt             46.05
## roll_dumbbell          43.77
## magnet_dumbbell_x      42.23
## magnet_belt_y          38.72
## magnet_belt_z          38.53
## accel_belt_z           29.40
## gyros_belt_z           28.47
## accel_dumbbell_y       26.69
## total_accel_belt       26.49
## magnet_arm_x           26.36
## roll_arm               26.17
## accel_dumbbell_z       25.97
## total_accel_dumbbell   24.44
## accel_forearm_x        22.99
## yaw_dumbbell           22.22
```




```
##  [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
```