Practical Machine Learning

Rhay

4/16/2021

Background

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

Data The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har.

Data Processing

```
#Download the data and read into dataframes
trainUrl <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
trainFile <- "./data/pml-training.csv"
testFile  <- "./data/pml-testing.csv"
if (!file.exists("./data")) {
  dir.create("./data")
}
if (!file.exists(trainFile)) {
  download.file(trainUrl, destfile=trainFile, method="curl")
}
if (!file.exists(testFile)) {
  download.file(testUrl, destfile=testFile, method="curl")
}
trainRaw <- read.csv("./data/pml-training.csv")
testRaw <- read.csv("./data/pml-testing.csv")
dim(trainRaw)
```

```
## [1] 19622   160
```

```
dim(testRaw)
```

```
## [1]  20 160
```

The datasets contain 160 variable and 19622 observations in the training set and 20 in the test set. Convert columns to numeric data type and remove columns with large number NA's to work better with model.

```
#convert to numeric datatype
#trainRaw[, 7:159] <- lapply(trainRaw[,7:159], as.numeric)
#testRaw[, 7:159] <- lapply(testRaw[,7:159], as.numeric)
#remove columns with NAs
sum(complete.cases(trainRaw))
```

```
## [1] 406
```

```
trainRaw <- trainRaw[, colSums(is.na(trainRaw)) == 0]
testRaw <- testRaw[, colSums(is.na(testRaw)) == 0]
#remove columns with headers containing 'timestamp, window' as these aren't useful in model; convert re
#to numeric datatype
classe <- trainRaw$classe
trainRemove <- grepl("^X|timestamp|window", names(trainRaw))
trainRaw <- trainRaw[, !trainRemove]
trainCleaned <- trainRaw[, sapply(trainRaw, is.numeric)]
trainCleaned$classe <- classe
testRemove <- grepl("^X|timestamp|window", names(testRaw))
testRaw <- testRaw[, !testRemove]
testCleaned <- testRaw[, sapply(testRaw, is.numeric)]
```

The clean data sets contain 53 variables with 19622 observations for training and 20 for testing. Divide the data into a 70/30 split to create training and test sets

```
set.seed(1234)
inTrain <- createDataPartition(trainCleaned$classe, p=0.70, list=F)
crossValidation <- trainCleaned[inTrain, ]
crossValidationTest <- trainCleaned[-inTrain, ]
```

Random Forest Model

Use Random Forest model to help determine which variables are most important. I choose this model as it is flexible, easy to use and known to give good results most of the time. I use a 5-fold cross validation for the algorithm.

```
controlRf <- trainControl(method="cv", 5)
modelRf <- train(classe ~ ., data=crossValidation, method="rf",trControl=controlRf, ntree=250)
modelRf
```

```
## Random Forest
##
```

```
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10989, 10991, 10989, 10991, 10988
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9903906  0.9878430
##   27    0.9910463  0.9886730
##   52    0.9838393  0.9795553
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

Test accuracy of random forest model on cross validation test data set

```
pred1 <- predict(modelRf, crossValidationTest)
#confusionMatrix(testData$classe, predictRf)
ClasseFactor<- as.factor(crossValidationTest$classe)
confusionMatrix(pred1, ClasseFactor)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    4    0    0    0
##          B    0 1130   14    0    0
##          C    1    5 1010    8    1
##          D    0    0    2  955    0
##          E    0    0    0    1 1081
##
## Overall Statistics
##
##                Accuracy : 0.9939
##                  95% CI : (0.9915, 0.9957)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9923
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9921   0.9844   0.9907   0.9991
## Specificity            0.9991   0.9971   0.9969   0.9996   0.9998
## Pos Pred Value         0.9976   0.9878   0.9854   0.9979   0.9991
## Neg Pred Value         0.9998   0.9981   0.9967   0.9982   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
```

```
## Detection Rate         0.2843   0.1920   0.1716   0.1623   0.1837
## Detection Prevalence   0.2850   0.1944   0.1742   0.1626   0.1839
## Balanced Accuracy      0.9992   0.9946   0.9907   0.9951   0.9994
```

```
accuracy <- sum(pred1 == ClasseFactor) / length(pred1)
cat('\n','The accuracy is',accuracy,'\n')
```

```
##
##  The accuracy is 0.9938828
```

```
outofsampleerror <- 1 - as.numeric(confusionMatrix(ClasseFactor, pred1)$overall[1])
cat('The out of sample error is',outofsampleerror,'\n')
```

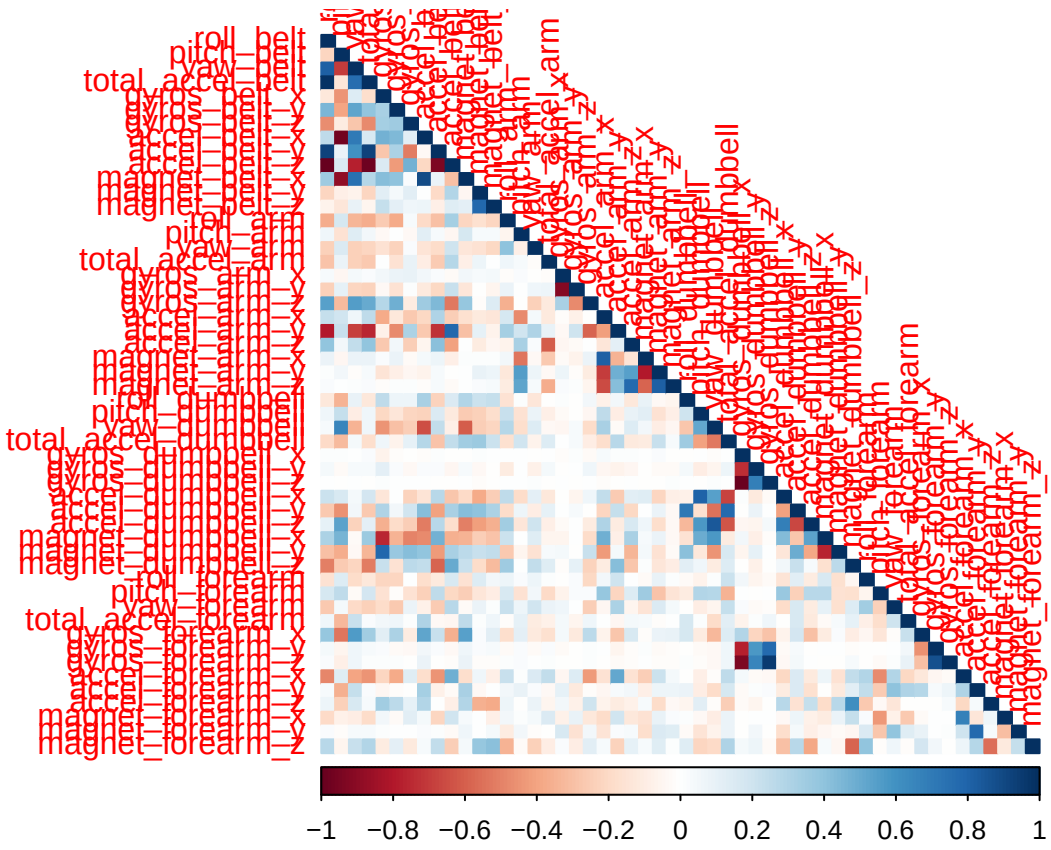
```
## The out of sample error is 0.006117247
```

The model accuracy proved to be good at 99.39% with an out of sample error of 0.61%. Based on this result, the random forest model appears to be good enough to make the necessary prediction.

Plots from Model

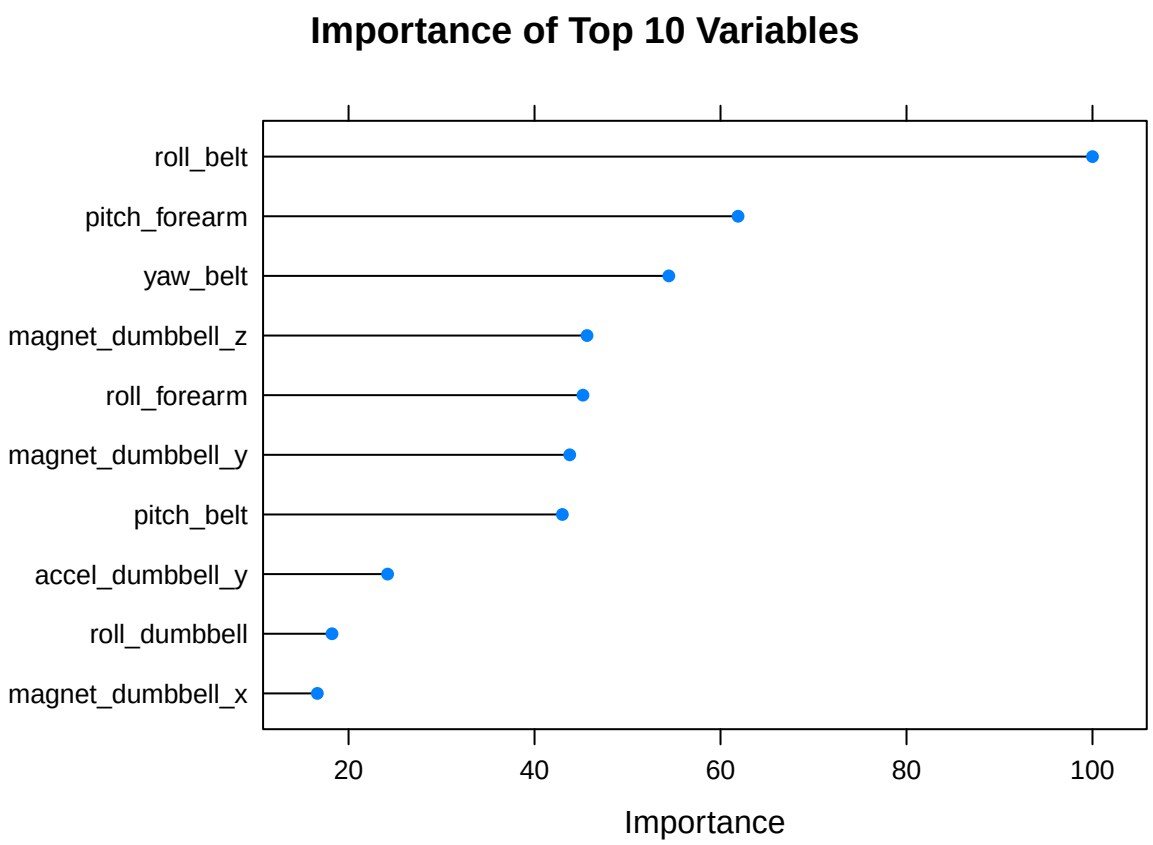
The Correlations matrix is below but it's hard to interpret due to the number of variables. In general, a darker color indicates more correlation between the two variables.

```
corrPlot <- cor(crossValidation[, -length(names(crossValidation))])
corrplot(corrPlot, method = "color", type="lower")
```



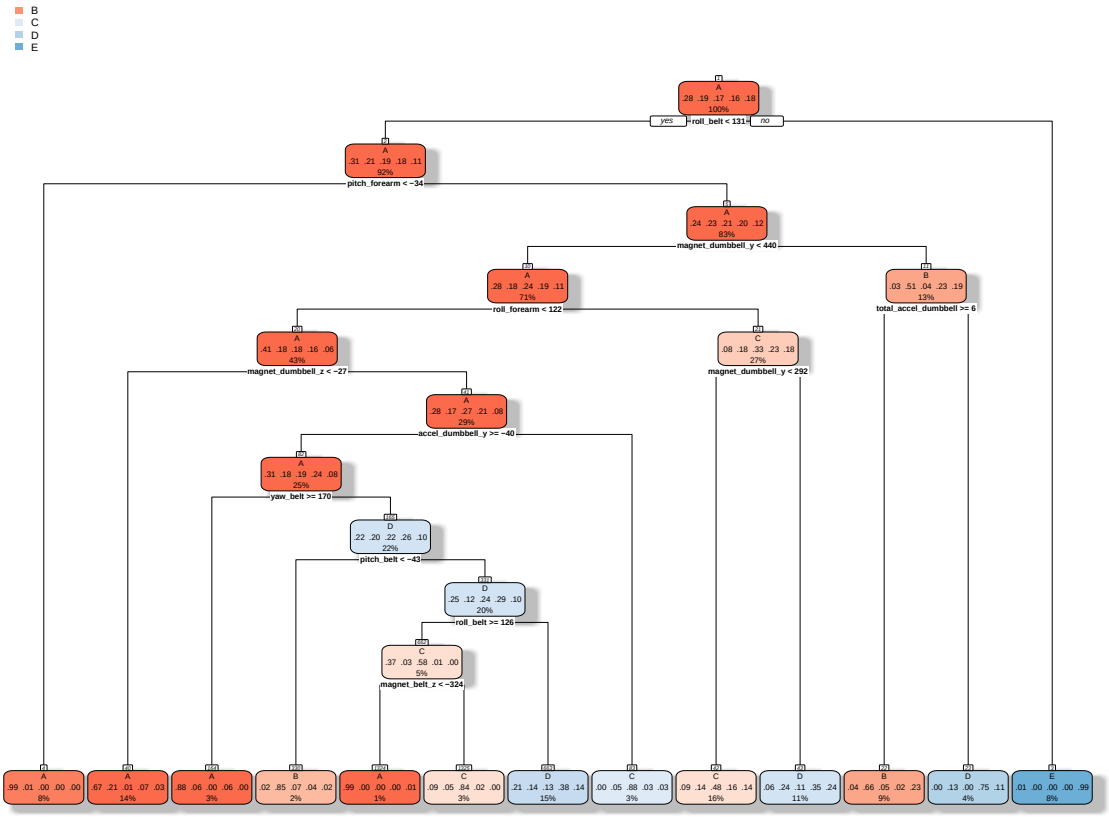
From the chart below it's easier to see the top 10 variables the model determined to be of most importance. For example, we see that the roll_best is the most important variable.

```
varImpObj <- varImp(modelRf)
plot(varImpObj, main = "Importance of Top 10 Variables", top = 10)
```



Another way to view which variables the model found to be most important is with a decision tree shown below.

```
tree2 <- rpart(classe ~ ., data=crossValidation, method="class", cp=.02)
rpart.plot(tree2, box.palette = "RdBu", shadow.col = "gray", nn=TRUE)
```



Test Model

To test the model, we predict test data using the random forest model we've developed.

```
result <- predict(modelRf, testCleaned[, -length(names(testCleaned))])
result
```

```
##  [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
```

Prepare model for submission to Coursera course

```
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)
}
}
x <- testRaw

answers <- predict(modelRf, newdata=x)
answers
```

```
##  [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
```

```
pml_write_files(answers)
```