Practical Learning Machine Assignment

Rohidah Maskuri

November 20, 2015

Executive Summary

In this project, our 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. It will then predict the manner in which they did the exercise. This is the "classe" variable in the training set. This report will describe how we built our model, how we used cross validation, what we think the expected out of sample error is, and why we made the choices you did. We will also use our prediction model to predict 20 different test cases.

Splitting the Data

Since the data already been splitted into the testing set and training set, both the files are loaded separately. As we can see, there are 19622 rows and 160 columns of data available form the training data set and 20 rows and 160 columns of testing data set.

```
setwd("C:/Users/rohidah/Google Drive/3  Data Science/Practical-Machine-
Learning")
testingori <- read.csv("pml-testing.csv", na.strings=c("NA",""),
strip.white=T)
training <- read.csv("pml-training.csv", na.strings=c("NA",""),
strip.white=T)
dim(training)

## [1] 19622   160

dim(testingori)

## [1]  20 160
```

Trimming the variables

Within the 160 variables, there are a lot of variables that will not contribute to the analysis. In this section, we are removing all those columns that contain "NA" and has no numeric value from the training data set. This leaves us with only 60 variables to review.

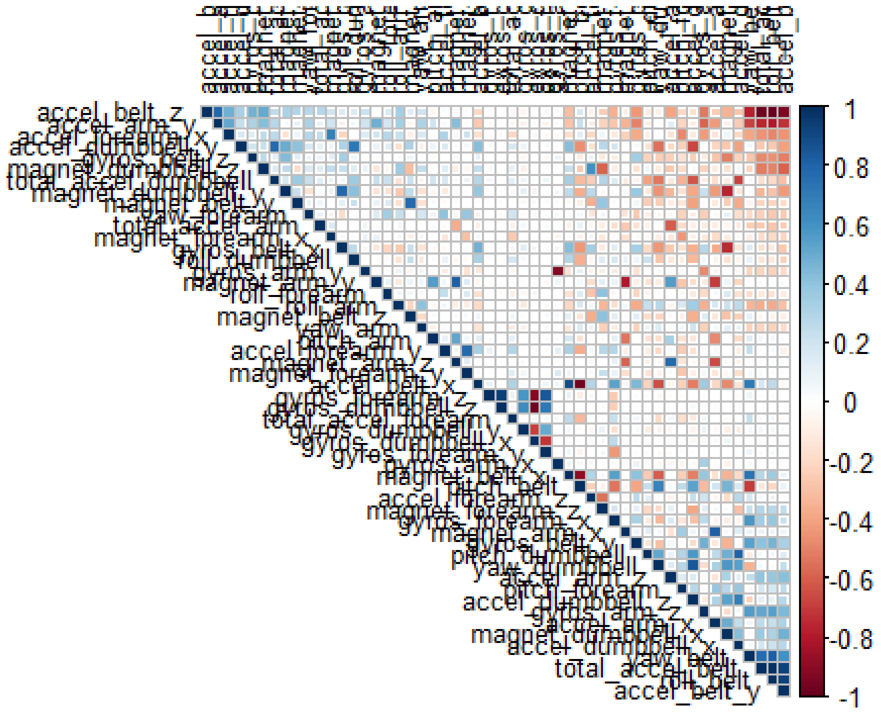
```
training2 <- training[ , colSums(is.na(training)) == 0]
dim(training2)

## [1] 19622    60
```

When we review the variables, the first 7 variables are information that are not significant for our analysis. Thus we are removing those 7 non-relevant variables. This process will reduce the variables further to 53 variables with the last variable being classe, the variable that we are predicting.

```r
training3<-training2[,-c(1:7)]
dim(training3)

## [1] 19622    53
```

The last step in preparing the baseline training data is to look at the correlations between the variables in our dataset. We may want to remove highly correlated predictors from our analysis and replace them with weighted combinations of predictors. This may allow a more complete capture of the information available.

```r
library(corrplot)
corrmatrix<- cor(na.omit(training3[sapply(training3, is.numeric)]))
dim(corrmatrix)

## [1] 52 52

corrplot(corrmatrix, order = "FPC", method = "square", type = "upper", tl.cex
= 0.8, tl.col = rgb(0, 0, 0))
```



Plotting the correlation

```r
library(caret)
```

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

This grid shows the correlation between pairs of the predictors in our dataset. From a high-level perspective darker blue and darker red squares indicate high positive and high negative correlations, respectively. Those that are highly correlated will be be removed with a a cutoff determined at 90%.Removing the highly correlated variable (90% and above)

```
removecor = findCorrelation(corrmatrix, cutoff = .99, verbose = TRUE)

## Compare row 10  and column  1 with corr  0.992 
##   Means:  0.27 vs 0.168 so flagging column 10 
## All correlations <= 0.99

training5 = training3[,-removecor]
dim(training5)

## [1] 19622    52
```

We now split the updated training dataset into a training dataset (70% of the observations) and a validation dataset (30% of the observations). This validation dataset will allow us to perform cross validation when designing our model.```

Cross Validation - Splitting data to test and train

```
inTrain <- createDataPartition(y=training5$classe, p=0.7, list=FALSE)
training <- training5[inTrain,]
testing <- training5[-inTrain,]
dim(training);dim(testing)

## [1] 13737    52

## [1] 5885   52
```

Random Forest Model & Cross Validation Testing

Next, we train a model using a random forest approach on the smaller training dataset.

```
library(randomForest)

## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.

modFitB1 <- randomForest(classe ~. , data=training)
predictionsB1 <- predict(modFitB1, testing, type = "class")
confusionMatrix(testing$classe,predictionsB1)

## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
```

```
##          B    5 1132    2    0    0
##          C    0    2 1021    3    0
##          D    0    0    8  955    1
##          E    0    0    0    7 1075
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9952          
##                  95% CI : (0.9931, 0.9968)
##     No Information Rate : 0.2853          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.994           
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9970   0.9982   0.9903   0.9896   0.9991
## Specificity            1.0000   0.9985   0.9990   0.9982   0.9985
## Pos Pred Value         1.0000   0.9939   0.9951   0.9907   0.9935
## Neg Pred Value         0.9988   0.9996   0.9979   0.9980   0.9998
## Prevalence             0.2853   0.1927   0.1752   0.1640   0.1828
## Detection Rate         0.2845   0.1924   0.1735   0.1623   0.1827
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9985   0.9984   0.9946   0.9939   0.9988
```

Accuracy and Out-of-Sample Error Estimate

Random forest trees were generated for the training dataset using cross-validation. Then the generated algorithm was examnined under the partitioned training set to examine the accuracy and estimated error of prediction. By using 52 predictors for five classes using cross-validation an accuracy of 0.9958 with a 95% CI [(0.9937, 0.9972)] was achieved accompanied by a Kappa value of 0.9946. Thus the The estimated accuracy of the model is 99.58% and the estimated out-of-sample error based on our fitted model applied to the cross validation dataset is 0.42%.

Predicted Results

Finally, we We then run our model against the testing dataset and display the predicted results

```r
testingori<-testingori[,names(training[,-52])]
predictionsB2<-predict(modFitB1,testingori, type="class")
predictionsB2
```

```
##  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
```