Practical Machine Learning: Course Project

Sizwe Mashao

14 June 2017

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. 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).

The 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

Cleaning the Data

To to Clean the data, wrote a function that removes empty columns and columns that dont have any variation. It also removes the first six coloumns(names, dates and stuff). The function can be seen in the code chunk below.

```r
clean.data <- function(data,thresh_hold = 0.5)
{
require(dplyr)
require(caret)
xx <- as.data.frame(sapply(data,is.na))
xy <- as.data.frame(sapply(xx,sum))/nrow(xx)
xz <- subset(xy,sapply(xx, sum)/nrow(xx) <= thresh_hold )
xs <- as.data.frame(t(xz))
xs <- subset(data, select = names(xs))
nsv <- nearZeroVar(xs, saveMetrics = T)
nsv <- subset(nsv, nsv$nzv =="FALSE")
nmes <- names(as.data.frame(t(nsv)))
subset(xs,select = nmes)[c(-1:-6)]
}
```

The Model

Now we can download and model the data. The data is split into a training and test set (70:30) and then cleaned as per above.

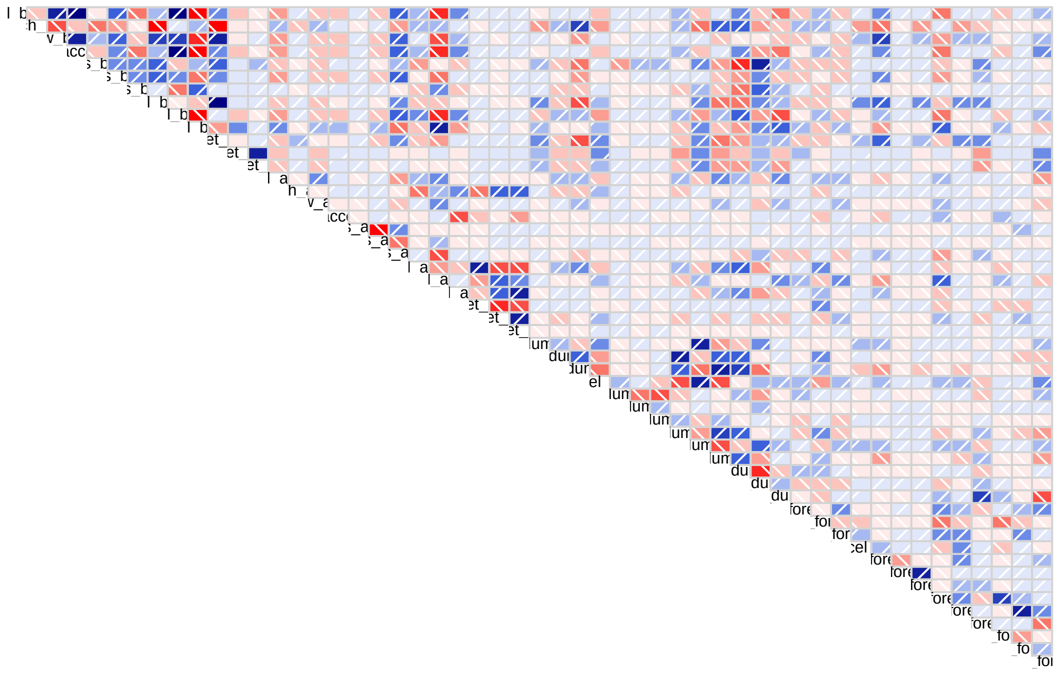
```
train_url <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
test_url <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

training <- read.csv(train_url)
test_final <- read.csv(test_url)

in_train  <- createDataPartition(training$classe, p=0.7, list=FALSE)
train <- training[in_train, ]
test  <- training[-in_train, ]

train <- clean.data(train)
test <- clean.data(test)
```

Before running the models we fit a correlogram to the data to see if any correlations stand out



Now we can fit the models. We fit a Random forest and gbm model to see which is more accurate.

```
cv_mod <- trainControl(method="cv", number=3, verboseIter=FALSE)

garbage <- capture.output(fit_rf <- train(classe ~ ., data=train, method="rf",trControl=cv_mod))
garbage1 <- capture.output(fit_gbm <- train(classe ~ ., data=train, method="gbm",trControl=cv_mod))
predict_rf <- predict(fit_rf, newdata=test)
predict_gbm <- predict(fit_gbm, newdata=test)
```

```r
x <- confusionMatrix(predict_rf, test$classe)
y <- confusionMatrix(predict_gbm, test$classe)
```

Results

Now we check which the models was more accurate.

confusion Matrix for the random forest:

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1669   14    0    0    0
##          B    4 1124    9    0    0
##          C    0    1 1007    8    1
##          D    0    0   10  954    2
##          E    1    0    0    2 1079
## 
## Overall Statistics
## 
##                Accuracy : 0.9912
##                  95% CI : (0.9884, 0.9934)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9888
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9970   0.9868   0.9815   0.9896   0.9972
## Specificity            0.9967   0.9973   0.9979   0.9976   0.9994
## Pos Pred Value         0.9917   0.9886   0.9902   0.9876   0.9972
## Neg Pred Value         0.9988   0.9968   0.9961   0.9980   0.9994
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2836   0.1910   0.1711   0.1621   0.1833
## Detection Prevalence   0.2860   0.1932   0.1728   0.1641   0.1839
## Balanced Accuracy      0.9968   0.9920   0.9897   0.9936   0.9983
```

confusion Matrix for the gbm:

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1643   40    0    1    1
##          B   19 1073   33    1   12
##          C    6   25  978   31   10
##          D    4    0   12  923   12
##          E    2    1    3    8 1047
##
## Overall Statistics
##
##                Accuracy : 0.9624
##                  95% CI : (0.9573, 0.9672)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9525
##  Mcnemar's Test P-Value : 1.706e-05
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9815   0.9421   0.9532   0.9575   0.9677
## Specificity            0.9900   0.9863   0.9852   0.9943   0.9971
## Pos Pred Value         0.9751   0.9429   0.9314   0.9706   0.9868
## Neg Pred Value         0.9926   0.9861   0.9901   0.9917   0.9927
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2792   0.1823   0.1662   0.1568   0.1779
## Detection Prevalence   0.2863   0.1934   0.1784   0.1616   0.1803
## Balanced Accuracy      0.9858   0.9642   0.9692   0.9759   0.9824
```

from this we see that both are very accurate but the random forrest was more accurate.

Predictions

Now all thats left is to run the perdictions on the test set using the RF model.

```
##  [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
```

Thank you.