Predicting Body Postures and Movement through Wearables

Lenny Fenster

Friday, December 18, 2014

Executive Summary

Devices such as Jawbone Up, Nike FuelBand, and Fitbit make it is easy and inexpensive to collect a large amount of data about personal activity. One thing that people regularly do with these devices is quantify how much of a particular activity they perform. However, they rarely quantify how well they do it. In this project, we used data from accelerometers to predict the manner in which an exercise was performed.

We utilized a *random forest* machine learning algorithm via the *caret* package. This method proved to be very successful and resulted in a Kappa statistic of 0.9997 for the out-of-sample error and an accuracy of 99.98%.

Getting the data

The training data for this project was obtained from https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv. The data needed to perform the project tests was obtained from https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv. All the data for this project came from http://groupware.les.inf.puc-rio.br/har.

The first step in this project was to obtain the data from the sources listed above and store them. Upon examination of the data, we discovered that there were several N/A fields that were either blank, had values of NA, or values of #DIV/0!. Therefore, we read those in as NA upon importing the data.

```
library(caret)
```

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

```
gadata.orig<-read.csv("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
                      na.strings= c("", "NA", "#DIV/0!"))
```

Investigation showed that 100 of the variables in the data set consisted primarily of NA values. A function was written to go through the variables and remove any where it consisted of more than 90% NA. The function also removed the identifier field from the data set as this variable would erroneously skew the learning.

```
cleandata <- function(df)
{
    numCols<-ncol(df)
    i<-8
    while(i  < numCols) {
        if ((sum(is.na(df[,i]))/nrow(df)) > 0.9) {
            df[i]<-NULL
            numCols<-numCols-1
        }
        else i<-i+1
    }

    #remove the identifier column
    return (df[-1])
}
```

This function was then called to create a cleaned version of the data set.

```
gadata.clean<-cleandata(gadata.orig)
```

Cross-validation

Before a model was fit, a cross-validation technique was employed by splitting the data into training and testing sets. We use the training set to build the model and later evaluated the results on the testing set. The code below uses the *createDataPartition()* method in the *caret* package to do this.

```
set.seed(54321)
inTrain <- createDataPartition(gadata.clean$classe, p = 3/4, list=FALSE)
training <- gadata.clean[inTrain,]
testing<-gadata.clean[-inTrain,]
```

Fitting the model

With training and testing sets created, we employed the **random forest** method encapsulated within the *caret* package for our model.

```
set.seed(12345)
modelFit<-train(classe~.,method="rf", data=training)
```

Estimating Out-of-Sample Error

We then employed our cross-validation technique by predicting the values for our testing set and ascertaining the out-of-sample error with it. This demonstrated that the model fit very well for this data with a accuracy of 99.98%.

```
predictTest<-predict(modelFit, testing)
cm<-confusionMatrix(predictTest, testing$classe)
print(cm)
```

Conclusion/Epilogue

The last step in this project was to use this model to on twenty new trials. This data was downloaded with the same arguments as the original data (aside from the different URI) and the previously aforementioned function that was used to clean the orginal data was used to clean this one as well. These results were then submitted for the twenty new trials. All submissions were deemed correct.

```
validation<-read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",
                     na.strings= c("", "NA", "#DIV/0!"))
validation.clean<-cleandata(validation)
predictValidation<-predict(modelFit, validation.clean)
```