Machine Learning Project Analysis

Satyam Sawant

```
## Warning: package 'caret' was built under R version 4.0.2
```

This document summarizes an attempt to build a simple machine learning algorithm, intended to predict from a sample data set how a barbell lift was performed.

Preprocessing

The data was first read in using the read.csv() function:

```r
training <- read.csv("pml-training.csv")
```

The resulting data frame had 160 variables and 19,622 observations. The variables were rendered as one of three classes: numeric, integer, and factor. In addition, multiple columns had a number of entries where the contents were "" or NA. For consistency, all data columns were converted to numeric; at the same time, the first seven variables (index, user, timestamp, and window data) were discarded.

```r
## Create a vector with all of the columns' classes
columnClasses <- sapply(training,class)

## Convert all data columns to numeric
for(i in 8:159){
        if(columnClasses[i] != "numeric"){
            if(columnClasses[i] == "factor"){
                training[training[,i] == "",i] <- NA
                training[,i] <- as.numeric(as.character(training[,i]))
            }
            training[,i] <- as.numeric(training[,i])
        }
    }

## Remove the non-data columns
training <- training[,-c(1:7)]
```

This, in turn, generated a number of columns with NAs. In fact, any columns with NAs consisted almost entirely of NA entries;

```r
NAcount <- sapply(training, function(x) sum(is.na(x)))
unique(NAcount)
```

```
##  [1]     0 19226 19248 19622 19225 19216 19294 19296 19227 19293 19221 19218
## [13] 19220 19217 19300 19301 19299
```

The very small amount of data made any attempt to impute or otherwise fill in the data risky at best; as such, all columns with NA values were eliminated.

```
training <- training[,ifelse(NAcount == 0, TRUE, FALSE)]
```

Feature Selection

In order to select the relevant features, the various variables were plotted graphically. To begin, the variables were plotted a few at a time using boxplots in R's featurePlot() function, like so:

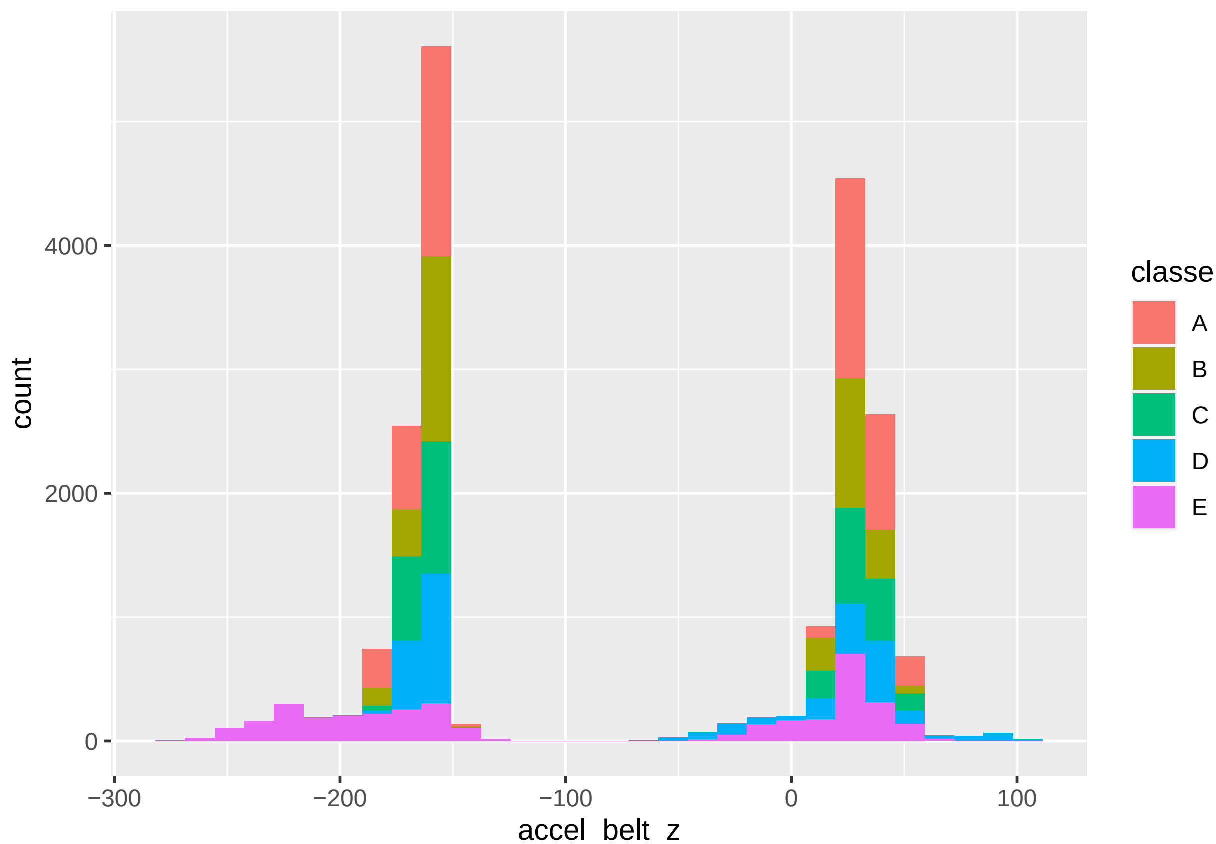
```
featurePlot(x = training[,8:10], y = training$classe, plot = "box")
```

```
## NULL
```

At this point, the plots were visually inspected. Any variable where the boxes for a single variable had some significant differences were then further inspected via a stacked histogram, such as this one for the accel_belt_z variable above:

```
ggplot(data = training, aes(x = accel_belt_z, fill = classe)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Those that were determined to be potentially useful were then noted, and ultimately used in the final model.

Cross validation

In order to properly cross-validate the model, the following procedure was repeated 10 times:

First, the training data was randomly split into two subsets - one for training, one for testing, like so:

```
splitter <- createDataPartition(training$classe, p = 0.6, list = FALSE)
training_train <- training[splitter,]
training_test <- training[-splitter,]
```

The training_train subset was then used to determine a model, using the features selected previously. The models were then used to predict the outcomes of each training_test subset. Finally, the accuracy of each prediction was determined and recorded down.

Once all 10 iterations were complete, the accuracies were averaged together to get a single estimate of the accuracy, from which an estimate of the out-of-sample error could be determined.

The Final Model

The final version of the model uses a total of seven variables: yaw_belt, accel_belt_z, gyros_arm_x, accel_arm_x, roll_dumbbell, gyros_dumbbell_x, and magnet_arm_x. The method used was the train() function's default method, the random forest.

The estimated accuracy of the model on the training_test subsets was 94%, indicating an out-of sample error of 6%.