A Machine Learning Prediction Algorithm for an Exercise Dataset

Author: Sukanya Basu

Synopsis

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, our goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to measure how well they exercise. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Preprocessing the data

First we load the data.

```
library(downloader)
download("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
, destfile="training.csv")
download("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",
 destfile="testing.csv")
training = read.csv("training.csv")
testing = read.csv("testing.csv")
```

Next we replace the missing values by zeros. We also drop the first seven columns of the dataset since they are not very useful for prediction purposes.

```
training[is.na(training)] <- 0
testing[is.na(testing)] <- 0
training <- training[,-c(1:7)]
testing <- testing[,-c(1:7)]
```

We divide the data into a training set and a validation set.

```
library(caret)
```

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

```
set.seed(125)
inTrain = createDataPartition(training$classe, p=0.7, list=FALSE)
training_data = training[ inTrain,]
validation_data = training[-inTrain,]
```

We extract only the numeric features of the training and testing datasets.

```
numeric_features = which(lapply(training_data, class) %in% c("numeric"))
training <- cbind(training_data$classe, training_data[, numeric_features])
testing <- testing[, numeric_features]
names(training)[1] <- "classe"
```

Identifying the important predictors

We first identify the near-zero-variance predictors as follows.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
nsv <- nearZeroVar(training, saveMetrics=TRUE)
head(nsv,5)
```

```
##                 freqRatio percentUnique zeroVar   nzv
## classe           1.469526    0.03639805   FALSE FALSE
## roll_belt        1.143791    8.09492611   FALSE FALSE
## pitch_belt       1.030534   12.25158332   FALSE FALSE
## yaw_belt         1.120787   13.05962000   FALSE FALSE
## max_roll_belt 1683.250000    1.07010264   FALSE  TRUE
```

Note that most of the values in the percentUnique column of nsv are less than 20 as shown below.

```
summary(nsv$percentUnique)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0364  1.1940  1.6020  5.6630  1.9440 86.7200
```

This suggests that most of the predictor variables have low variability in general. In this case, the predictors with freqRatio less than 20 have a smaller probability of being near-zero-variance predictors since they exhibit a lower level of skewness (see [2] page 4). We sort nsv in increasing order of freqRatio to identify the top five predictors with the least probability of being near-zero-variance predictors.

```
nsv2 <- data.frame(rownames(nsv),nsv$freqRatio, nsv$nzv)
names(nsv2) <- c("rownames", "freqRatio", "nzv")
z1 <- arrange(nsv2, freqRatio)
head(z1,5)
```

```
##           rownames freqRatio   nzv
## 1 gyros_dumbbell_z  1.021226 FALSE
## 2     gyros_belt_z  1.022763 FALSE
## 3 magnet_forearm_z  1.023256 FALSE
## 4       pitch_belt  1.030534 FALSE
## 5 gyros_dumbbell_x  1.036866 FALSE
```

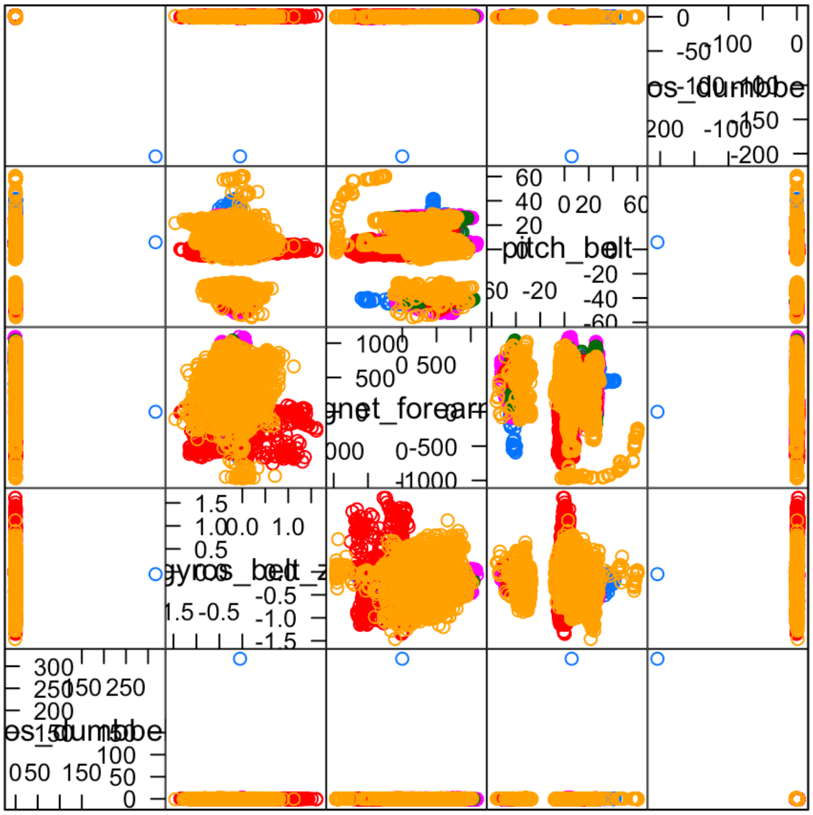
We present a feature plot and a correlation plot for these five predictors below.

```
library(caret)
library(psych)
```

```
##
## Attaching package: 'psych'
```

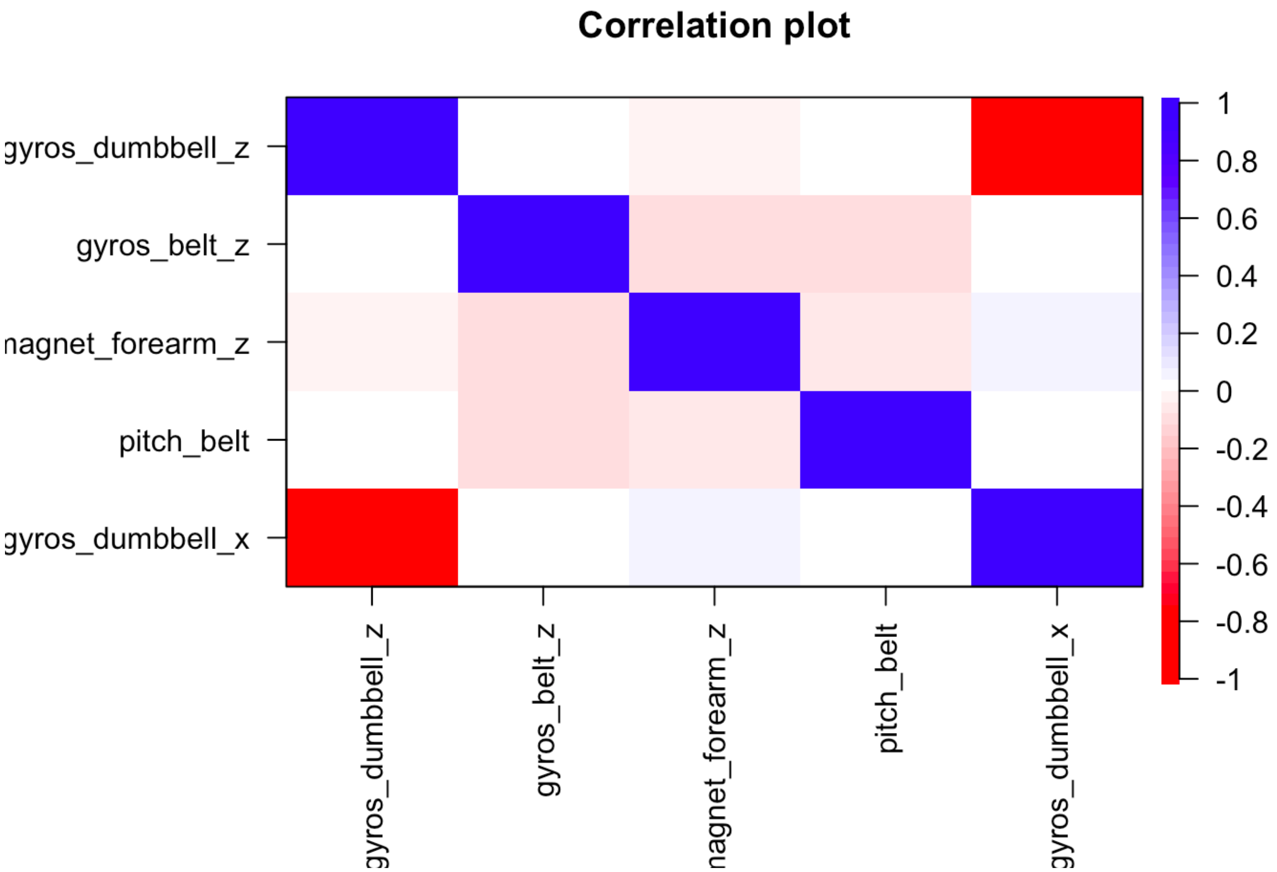
```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
featurePlot(x=training[,c("gyros_dumbbell_z", "gyros_belt_z",
                          "magnet_forearm_z", "pitch_belt", "gyros_dumbbell_x")
],
            y = training$classe, plot="pairs")
```



Scatter Plot Matrix

```
corPlot(training[,c("gyros_dumbbell_z", "gyros_belt_z",
                              "magnet_forearm_z", "pitch_belt", "gyros_dumbbell_x")
])
```



We find a very strong negative correlation between the predictors gyros_dumbbell_x and gyros_dumbbell_z as shown by the red boxes in the plot. We also find a faint positive correlation between the predictors gyros_dumbbell_x and magnet_forearm_z as shown by the pale blue boxes in the plot. There are also various levels of negative correlation between the predictors as shown by the salmon-colored boxes.

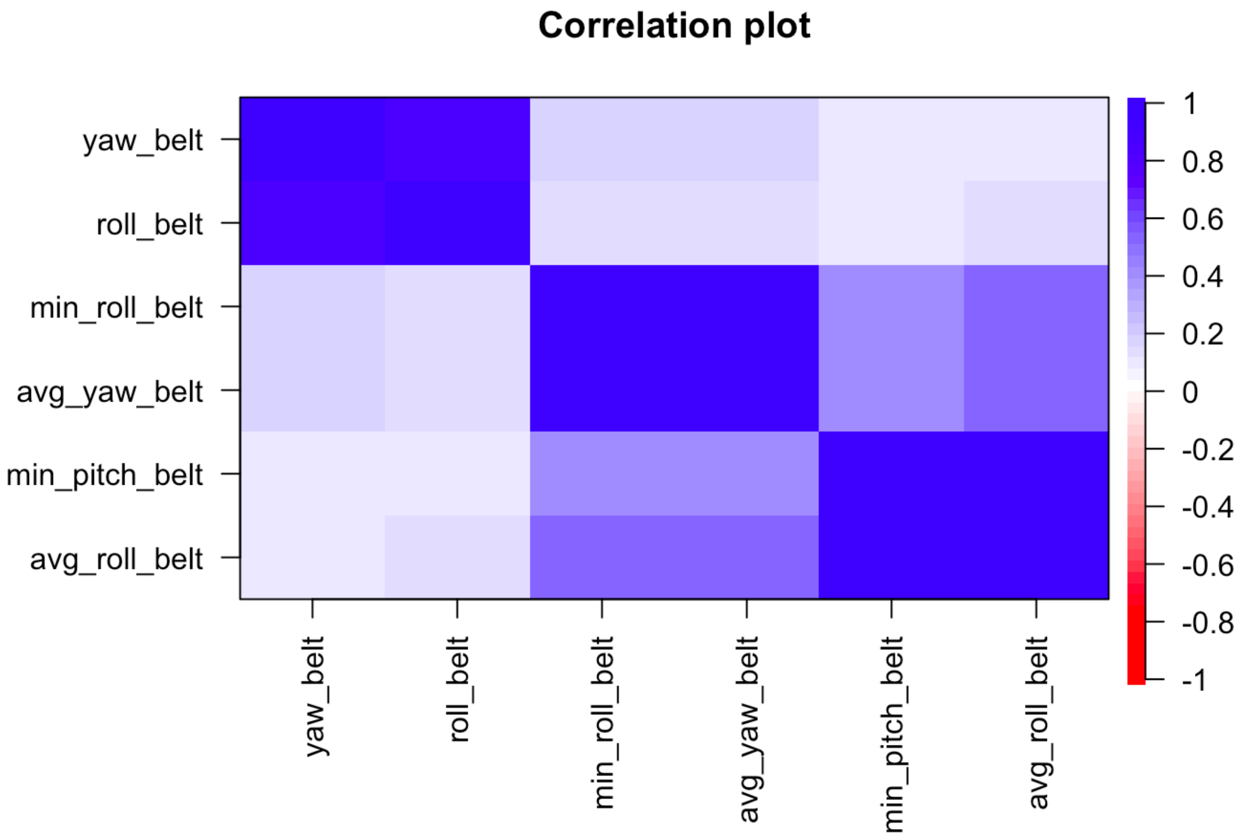
Identifying predictors with the highest correlation

Next we identify the predictors with the highest correlation and draw a correlation plot for them.

```
M <- abs(cor(training[,-1]))
diag(M) <- 0
head(which(M > 0.8, arr.ind=T))
```

```
##                row col
## yaw_belt         3   1
## roll_belt        1   3
## min_roll_belt    6   4
## avg_yaw_belt    17   4
## min_pitch_belt   7   5
## avg_roll_belt   11   5
```

```
library(psych)
corPlot(training[, c("yaw_belt", "roll_belt","min_roll_belt",
                    "avg_yaw_belt", "min_pitch_belt","avg_roll_belt")])
```



It is clear from the predominantly blue plot that all five variables are positively correlated to each other with varying strengths of correlation represented by the varying shades of blue.

Defining and cross-validating our Support Vector Machine (SVM) model

Now we are ready to define our Support Vector Machine (SVM) model to predict the ‘classe’ variable using all other numeric variables as predictors.

```
set.seed(325)
library(e1071)
svm_mod <- svm(classe ~ ., data=training)
svm_mod
```

```
##
## Call:
## svm(formula = classe ~ ., data = training)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.0106383
##
## Number of Support Vectors:  10167
```

Next we cross-validate our model using the validation data set.

```
predic <- predict(svm_mod, validation_data)
summary(predic)
```

```
##    A    B    C    D    E
## 1737  990 1209  969  980
```

Measuring the accuracy of our SVM model

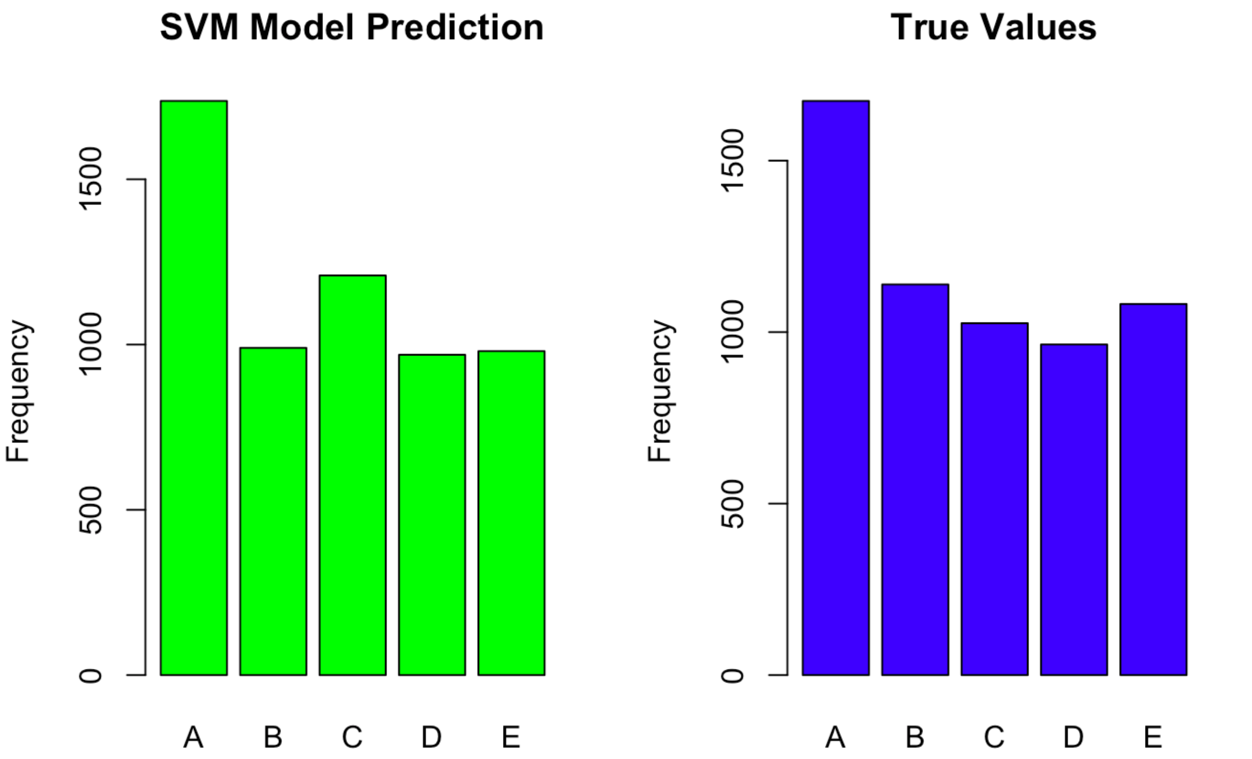
We measure the accuracy of our model by comparing the model prediction for the values of the ‘classe’ variable from the validation data with its true values.

```
confusionMatrix(predic,validation_data$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1522  136   48   15   16
##          B   24  802   74   16   74
##          C   41  117  797  142  112
##          D   46   40   88  741   54
##          E   41   44   19   50  826
## 
## Overall Statistics
##                                           
##                Accuracy : 0.7966          
##                  95% CI : (0.7861, 0.8068)
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.7426          
##  Mcnemar's Test P-Value : < 2.2e-16       
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9092   0.7041   0.7768   0.7687   0.7634
## Specificity            0.9489   0.9604   0.9152   0.9537   0.9679
## Pos Pred Value         0.8762   0.8101   0.6592   0.7647   0.8429
## Neg Pred Value         0.9634   0.9312   0.9510   0.9546   0.9478
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2586   0.1363   0.1354   0.1259   0.1404
## Detection Prevalence   0.2952   0.1682   0.2054   0.1647   0.1665
## Balanced Accuracy      0.9291   0.8323   0.8460   0.8612   0.8657
```

A comparison of the plots of the predicted and true values for the ‘classe’ variable from the validation data set is shown below.

```
par(mfrow = c(1,2))
plot(predic,col="green", ylab="Frequency",
     main="SVM Model Prediction")
plot(validation_data$classe, ylab="Frequency",col="blue",
     main="True Values")
title("A comparison of the predicted and true values for the 'classe' variable"
, line = -23.5, outer = TRUE)
```



A comparison of the predicted and true values for the 'classe' variable

Predicting test data using our SVM model

Finally we use our SVM model to make predictions on the testing data.

```
predict(svm_mod, testing)
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 
##  C  A  C  A  A  C  D  B  A  A  A  C  B  A  B  E  C  B  A  B 
## Levels: A B C D E
```

References

[1] Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13). Stuttgart, Germany: ACM SIGCHI, 2013.

[2] Kuhn, Max. Building Predictive Models in R Using the caret Package. Journal of Statistical Software. November 2008, Volume 28, Issue 5.