Human Activity Recognition Using Accelerometer Data

Dushan Yovetich

February 3, 2017

Introduction

As part of an independent study, 6 male participants between 20 and 28 years of age were asked to perform one set of 10 dumbell bicep curls using 5 different techniques - one correct technique and four common incorrect techniques. While performing the bicep curls, data was collected using accelerometers on the belt, forearm, arm and dumbell of each participant. Our goal is to predict which technique is used in performing the exercise based on the data.

Data

The data is provided courtesy of Groupware@LES. For further information regarding the data and research, refer to "Qualitative Activity Recognition of Weight Lifting Exercises" at the following link: http://groupware.les.inf.puc-rio.br/har. We read in our dataset.

```
# read in dataset
WLEdata <- read.csv(
        "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
        header = TRUE)
dim(WLEdata)
```

```
## [1] 19622   160
```

Each observation in the dataset has 160 variables including various measurements from the accelerometers. The 5 different lifting techniques to be predicted are captured in **classe**, a factor variable. The factor values are as follows:

Class	Value Description	Correct Lifting Technique
A	According to specification	Yes
B	Throwing elbows to front	No
C	Lifting elbows only halfway	No
D	Lowering dumbells only halfway	No
E	Throwing hips to the front	No

R Packages

```
#packages to use
library(caret)
library(randomForest)
library(parallel)
library(doParallel)
```

Transformations

As noted, there is a large population of covariates. We evaluate whether there are any zero covariates. These are variables we can consider eliminating from our model since they will likely not be good predictors due to their minimal variability. Further, if there are any columns with zero (i.e. N/A) values, we will need to remove those from our dataset as well. Otherwise, our model may not process correctly.

```
# Eliminate zero covariates
nsv <- nearZeroVar(WLEdata, saveMetrics = TRUE)
WLEdata <- subset(WLEdata,select = row.names(nsv[nsv$nzv==FALSE,]))

# Eliminate covariates with N/A
WLEdata <- WLEdata[,colSums(is.na(WLEdata))==0]
```

We are now left with 59 covariates. We review the structure of remaining covariates to see if there are any that could be further removed. There are some variables remaining with qualitative information, such as user name and belt number. Also, we have a series of timestamps captured during the study. These are not necessary for the purpose of our predictive model. Therefore, we can further remove those. We should now be left with a dataset of potentially useful quantitative predictor variables

```
# Eliminate qualitative covariates
WLEdata <- WLEdata[, -c(1:6)]
```

Model Building

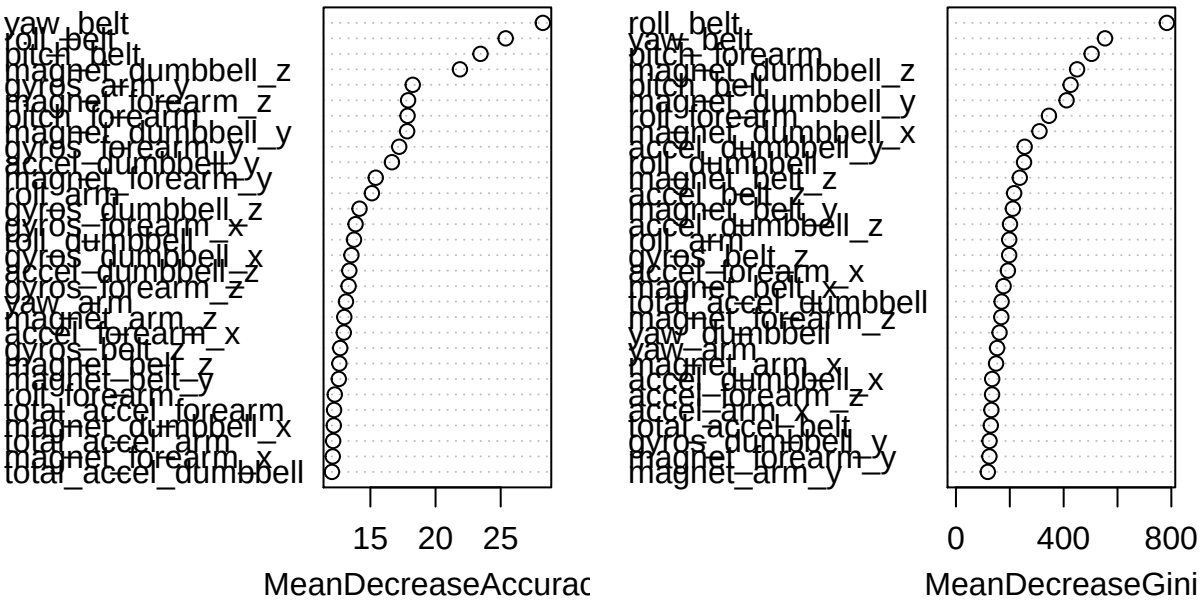
This appears to be a non-linear, classification problem. The most promising method is **random forest** as these are typically the most accurate and work well with classification predictions. Therefore, we will start here and evaluate whether we need to consider other methods.

Preprocessing

With our dataset, it could be computationally intensive to build a model using all our available covariates. We consider whether it would be possible to further limit the number of covariates. We start with some preprocessing with a random forest model and analyzing the importance of our covariates.

```
# Partition data into a training and testing dataset
set.seed(122)
inTrain <- createDataPartition(WLEdata$classe, p = .6, list = FALSE)
training <- WLEdata[inTrain,]
testing <- WLEdata[-inTrain,]

# build model
set.seed(121)
fit <- randomForest(classe~., data = training, importance=TRUE, ntree = 100)
varImpPlot(fit)
```



Based on our initial model, it appears we may be able to further limit the covariates in our model by their importance. We further subset our training set to **classe** and top ten covariates - based on Gini.

```
varTrain <- c("roll_belt","yaw_belt","pitch_forearm","magnet_dumbbell_z",
              "pitch_belt","magnet_dumbbell_y","roll_forearm","magnet_dumbbell_x",
              "magnet_belt_z","accel_dumbbell_y","classe")
training <- subset(training,select = varTrain)
```

Model Building

We are now ready to attempt to build our prediction model. We opt for a 3 k-fold cross validation for our random forest model. To hopefully speed the model building process, we also incorporate parallel processing, assigning n-1 cores for the task. Once we have our model, we will save it for future use.

```
if(file.exists("modFitRF.rda")){
  # load pre-existing model
  load("modFitRF.rda")
} else {
  # Set up cluster
  cluster <- makeCluster(detectCores() - 1)
  registerDoParallel(cluster)

  # cross validation settings
  trControl <- trainControl(method  = "cv", number = 3, allowParallel = TRUE)

  # Random Forest Model

  set.seed(1243)
  x <- training[,-11]
```

```
  y <- training[,11]
  modFitRF <- train(x,y, method = "rf", data = training, trainControl=trControl, prox = TRUE)
  save(modFitRF, file = "modFitRF.rda")

  # de-register cluster
  stopCluster(cluster)
  registerDoSEQ()
}
```

Model Accuracy

We now check to see how well the model is at predicting values in our testing set.

```
# Check accuracy
predRF <- predict(modFitRF,testing)
cm <-confusionMatrix(testing$classe,predRF)
print(cm)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2226    2    4    0    0
##          B    9 1503    5    1    0
##          C    0    6 1357    5    0
##          D    0    0    8 1278    0
##          E    0    3    5    0 1434
##
## Overall Statistics
##
##                Accuracy : 0.9939
##                  95% CI : (0.9919, 0.9955)
##     No Information Rate : 0.2849
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9923
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9960   0.9927   0.9840   0.9953   1.0000
## Specificity            0.9989   0.9976   0.9983   0.9988   0.9988
## Pos Pred Value         0.9973   0.9901   0.9920   0.9938   0.9945
## Neg Pred Value         0.9984   0.9983   0.9966   0.9991   1.0000
## Prevalence             0.2849   0.1930   0.1758   0.1637   0.1828
## Detection Rate         0.2837   0.1916   0.1730   0.1629   0.1828
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9975   0.9952   0.9912   0.9971   0.9994
```

Based on the confusion matrix, we have 0.9938822 accuracy using the ten covariates chosen. In conclusion, the model we've built is very accurate and we forgo any further modelling at this time.

Prediction

Now that we have a model, we can attempt to predict the outcome for a set of data not originally included in our training and testing set.

```
# read in dataset
predicting <- read.csv(
        "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",
        header = TRUE)

# transformations for dataset
nsv <- nearZeroVar(predicting, saveMetrics = TRUE)
predicting <- subset(predicting,select = row.names(nsv[nsv$nzv==FALSE,]))
predicting <- predicting[,colSums(is.na(predicting))==0]
predicting <- predicting[, -c(1:6)]
```

```
# predict outcomes
predicting$Outcome <- predict(modFitRF,newdata = predicting, type = "raw")
predicting$Outcome
```

```
##  [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
```