Machine Learning Project

treepruner

October 9, 2015

Project Description

Sensors were placed on the belt, forearm, arm and dumbell of 6 participants. These sensors detected the acceleration, gyroscope and magnetometer data at a joint sampling rate of 45 Hz. The participants were then asked to lift a barbell 5 different ways: * Class A exactly according to the specification * Class B throwing the elbows to the front * Class C lifting the dumbbell only halfway * Class D lowering the dumbbell only halfway * Class E throwing the hips to the front

Our task is to develop a model with only the accelerometer readings to identify which class method was used to lift the dumbell.

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har.

Executive Summary

The rpart method produced a model with an accuracy of XXX. The most useful variables proved to be

Load Packages

```
#if (!file.exists("project")) { dir.create("project")}
#setwd("./project")

#setInternet2(use = TRUE)
#trainURL <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
#download.file(trainURL, "pml_training.csv", method = "auto")

#testURL <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
#download.file(testURL, "pml_testing.csv", method = "auto")
```

Download Data

Load Training Data Set and Check Control Totals Control Totals per Forum: | A| B| C| D| E| |=====||=====||=====||=====||=====| | 5580| 3797| 3422| 3216| 3607|

My Training Totals:

```
pml_training <- read.csv("pml_training.csv", na.strings = c("#DIV/0!", "NA"))
table(pml_training$classe)
```

```
##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

Subset for classe and _X, _Y, _Z Variables Only Exclude username, timestamps, windows and classe variables Exclude all the variables calculated from the original sensor measurements identified with variable names containing accel, gyros, magnet and _x, _y or _z.

```
xyz <- select(pml_training, classe, contains("_x"), contains("_y"), contains( "_z"), -contains("stddev"
```

```
corXYZ <- abs(cor(xyz[, -1]))
diag(corXYZ) <- 0
corXYZ <- which(corXYZ > 0.8, arr.ind = T)
corXYZ[order(rownames(corXYZ)), ]
```

Explore _xyz Data with Correlations

```
##                  row col
## accel_arm_x        5   6
## accel_belt_x       2   3
## accel_belt_y      14  26
## accel_belt_z      26  14
## gyros_arm_x        4  16
## gyros_arm_y       16   4
## gyros_dumbbell_x   7  31
## gyros_dumbbell_x   7  34
## gyros_dumbbell_z  31   7
## gyros_dumbbell_z  31  34
## gyros_forearm_y   22  34
## gyros_forearm_z   34   7
## gyros_forearm_z   34  22
## gyros_forearm_z   34  31
## magnet_arm_x       6   5
## magnet_arm_y      18  30
## magnet_arm_z      30  18
## magnet_belt_x      3   2
```

```
xyz$accel_belt_xy <- xyz$accel_belt_x + xyz$accel_belt_y
xyz$gyros_arm_xy <- xyz$gyros_arm_x  + xyz$gyros_arm_y
xyz$gyros_dumbbell_xz <- xyz$gyros_dumbbell_x + xyz$gyros_dumbbell_z
xyz$gyros_forearm_yz <- xyz$gyros_forearm_y + xyz$gyros_forearm_z
xyz$magnet_arm_xyz <- xyz$magnet_arm_x + xyz$magnet_arm_y + xyz$magnet_arm_z
xyz$accelMagnet_belt_x <- xyz$accel_belt_x + xyz$magnet_belt_x
```

Combine Highly Correlated Variables

```
xyz <- select(xyz,
              -accel_belt_x, -accel_belt_y,
```

```
              -gyros_arm_x, -gyros_arm_y,
              -gyros_dumbbell_x, -gyros_dumbbell_z,
              -gyros_forearm_y, -gyros_forearm_z,
              -magnet_arm_x, -magnet_arm_y, -magnet_arm_z,
              -accel_belt_x,-magnet_belt_x)
```

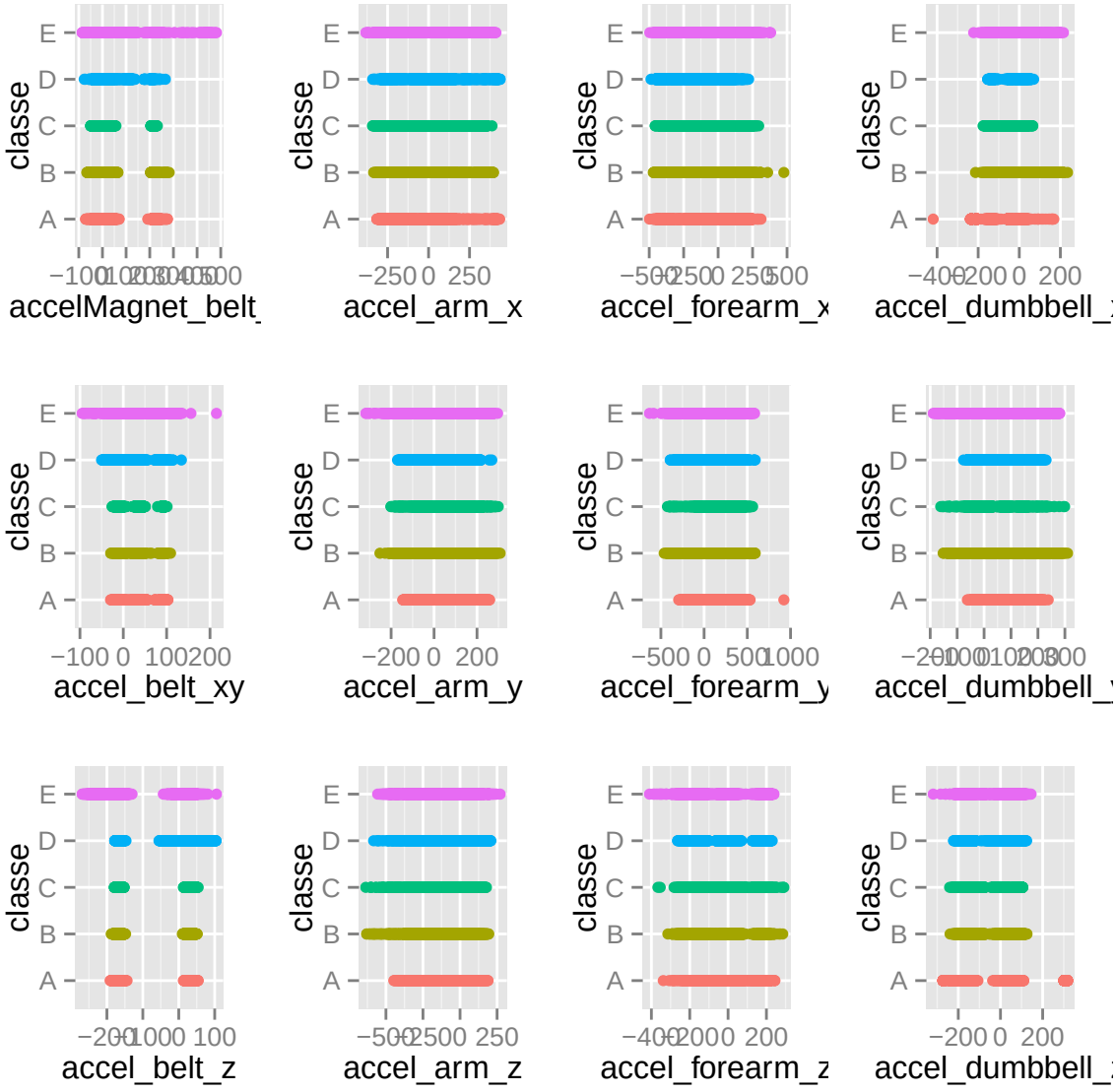
Remove Highly Correlated Variables

Create _xyz Training and Validation Set

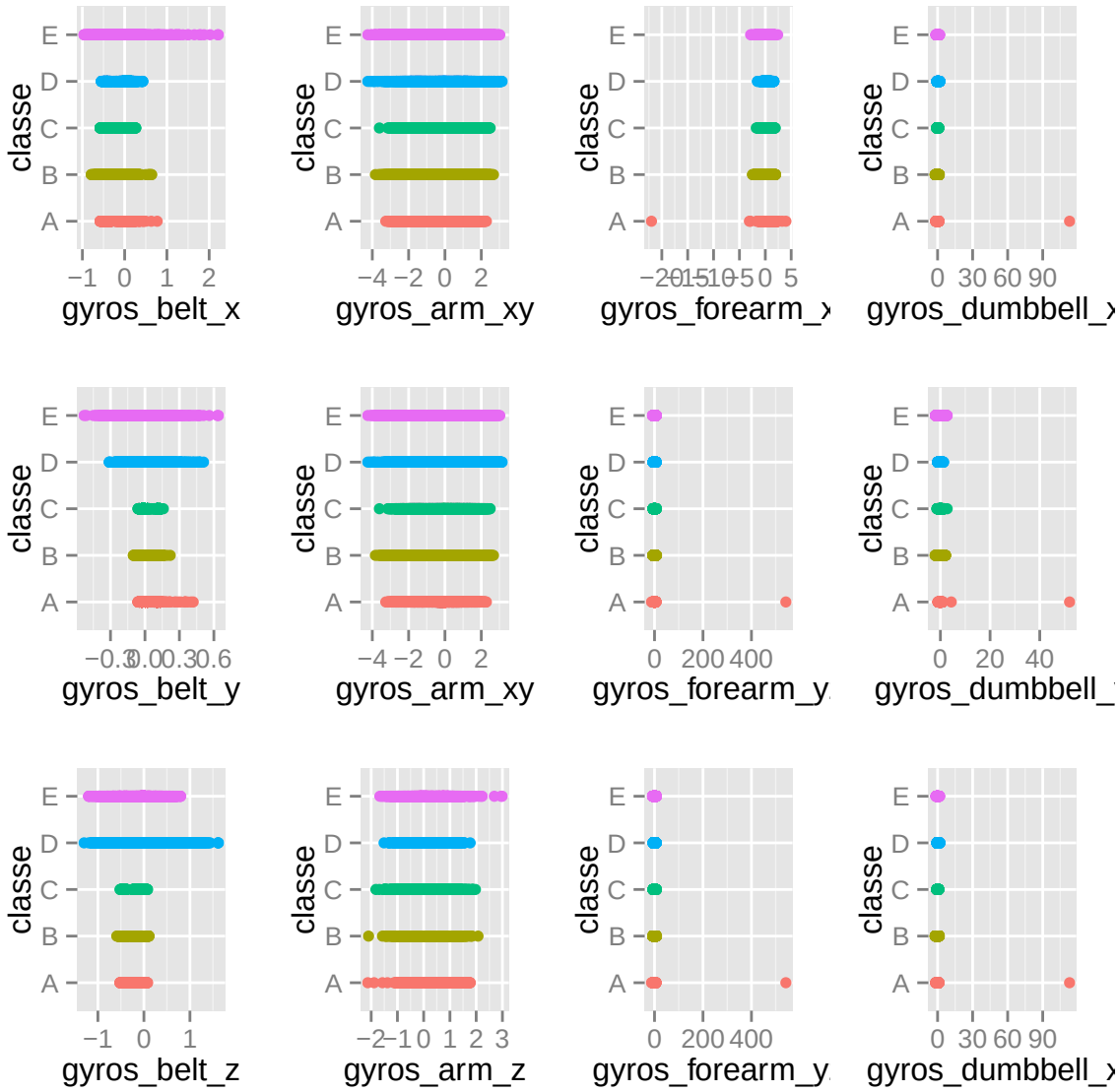
I used the recommended value of .6 to create a training set which leaves 40% for a validation set in order to calculate an out of sample error rate.

```
set.seed(1138)
inTrain_xyz <- createDataPartition(y = xyz$classe, p = 0.6, list = FALSE)
training_xyz <- xyz[inTrain_xyz,]
validation_xyz <- xyz[-inTrain_xyz,]
```

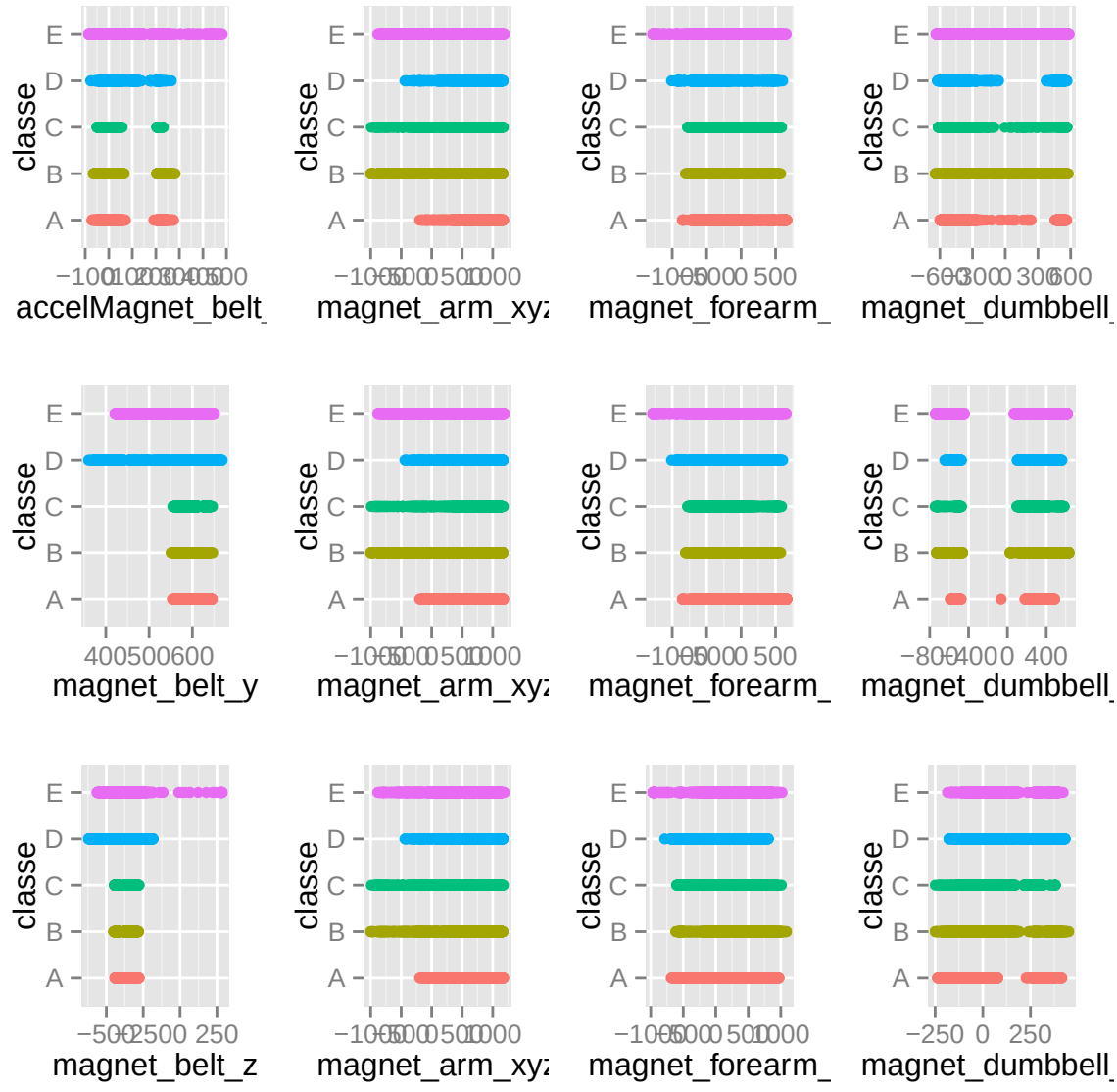
Explore _xyz Acceleration Variables Graphically



Explore _xyz Gyroscope Variables Graphically



Explore _xyz Magnetometer Variables Graphically



Run rpart xyz

```
trCtrl <- trainControl(method="repeatedcv",
                       repeats=5,
                       classProbs=TRUE) # accuracy : 0.4842

set.seed(1138)
registerDoParallel(4)
getDoParWorkers()
```

```
## [1] 4
```

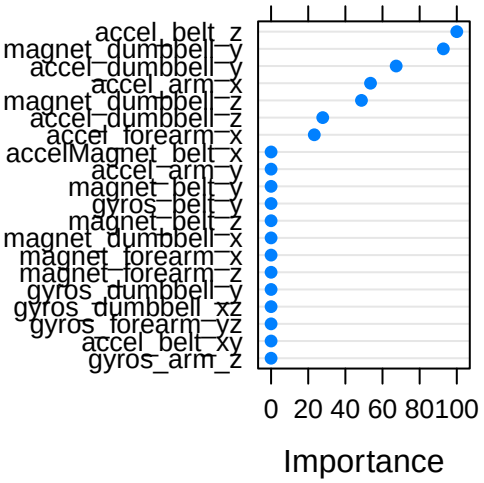
```
xyzFit <- train(classe ~., data = training_xyz,
             method = "rpart",
             control = rpart.control(minsplit = 2),
             trControl = trCtrl,
             tuneLength = 5)
```

```
## Loading required package: rpart
```

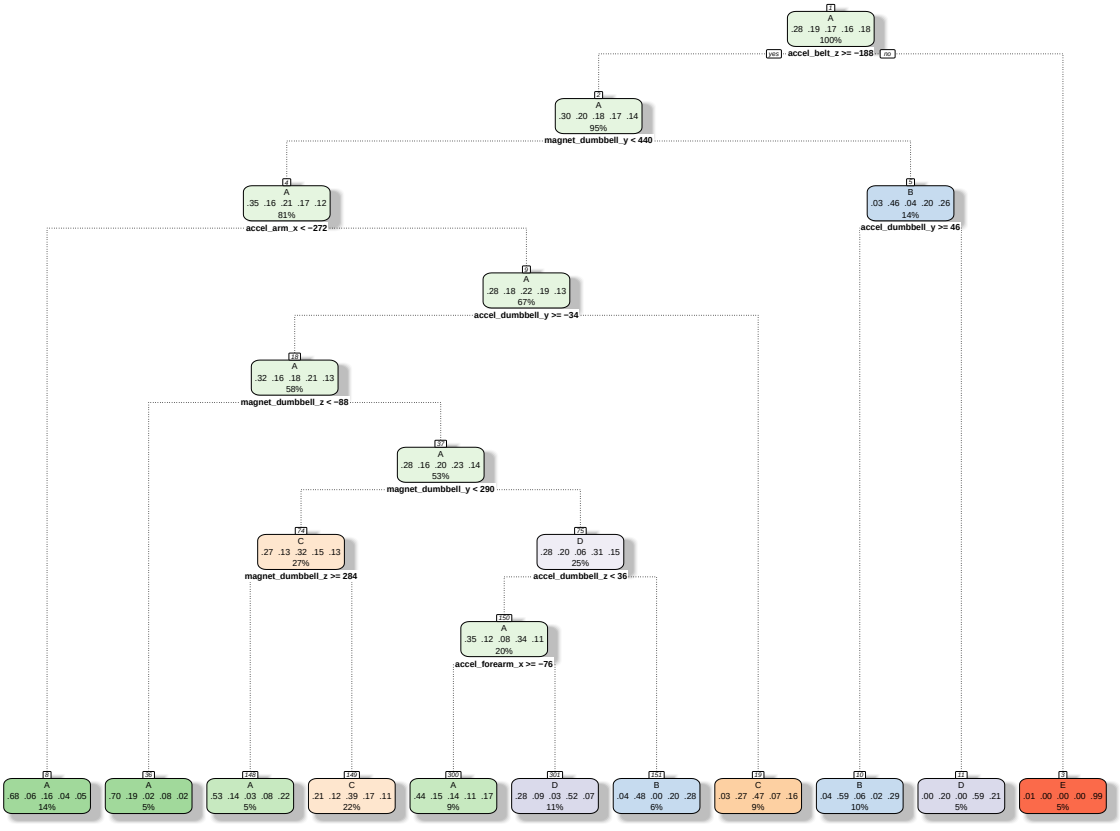
```
varImp(xyzFit, compete= FALSE)
```

```
## rpart variable importance
##
##   only 20 most important variables shown (out of 30)
##
##                   Overall
## accel_belt_z       100.00
## magnet_dumbbell_y   92.76
## accel_dumbbell_y    67.36
## accel_arm_x         53.55
## magnet_dumbbell_z   48.67
## accel_dumbbell_z    27.81
## accel_forearm_x     23.25
## gyros_belt_x         0.00
## magnet_forearm_z     0.00
## magnet_arm_xyz       0.00
## gyros_dumbbell_xz    0.00
## magnet_belt_z        0.00
## gyros_arm_xy         0.00
## gyros_forearm_yz     0.00
## accel_arm_y          0.00
## accel_arm_z          0.00
## magnet_belt_y        0.00
## accel_forearm_z      0.00
## magnet_forearm_x     0.00
## gyros_belt_z         0.00
```

```
dotPlot(varImp(xyzFit, compete= FALSE))
```




```
fancyRpartPlot(xyzFit$finalModel, sub="")
```



```
xyzFitPred <- predict(xyzFit, newdata = validation_xyz)
confusionMatrix(xyzFitPred, validation_xyz$classe)
```

Compare with Valadation xyz

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1604  282  260  185  253
##          B   40  651   36   99  326
##          C  354  420 1042  347  288
##          D  233  165   30  655  157
##          E    1    0    0    0  418
##
## Overall Statistics
##
##                Accuracy : 0.557
##                  95% CI : (0.5459, 0.568)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4376
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7186  0.42885   0.7617  0.50933  0.28988
## Specificity            0.8254  0.92083   0.7825  0.91082  0.99984
## Pos Pred Value         0.6207  0.56510   0.4251  0.52823  0.99761
## Neg Pred Value         0.8807  0.87048   0.9396  0.90448  0.86212
## Prevalence             0.2845  0.19347   0.1744  0.16391  0.18379
## Detection Rate         0.2044  0.08297   0.1328  0.08348  0.05328
## Detection Prevalence   0.3293  0.14683   0.3124  0.15804  0.05340
## Balanced Accuracy      0.7720  0.67484   0.7721  0.71008  0.64486
```

Load Testing Data Set

```
pml_testing <- read.csv("pml_testing.csv",
                        na.strings = c("#DIV/0!", "NA"))
dim(pml_testing)
```

```
## [1]  20 160
```