Machine Learning Project

treepruner

October 25, 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. I have interpreted this to mean all the original sensor readings.

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

Executive Summary

The overall accuracy when used against the validation data set was 0.557 with a Kappa of 0.4376. The model successfully predicted only 7 out of 20 test cases. The most useful variables for classification in this model proved to be:

- 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

The sensitivity, which is the TP / (TP + FN) for detecting Class A is 0.7186. The specificity, which is TN / (FN + TN), for Class A is .8254.

This model needs improvement.

```
library(dplyr)
library(grid)
library(gridExtra)
library(caret)
library(rattle)
library(party)
library(partykit)
library(doParallel)
library(pander)
```

Load Packages

Get and Clean Data

```
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 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)
```

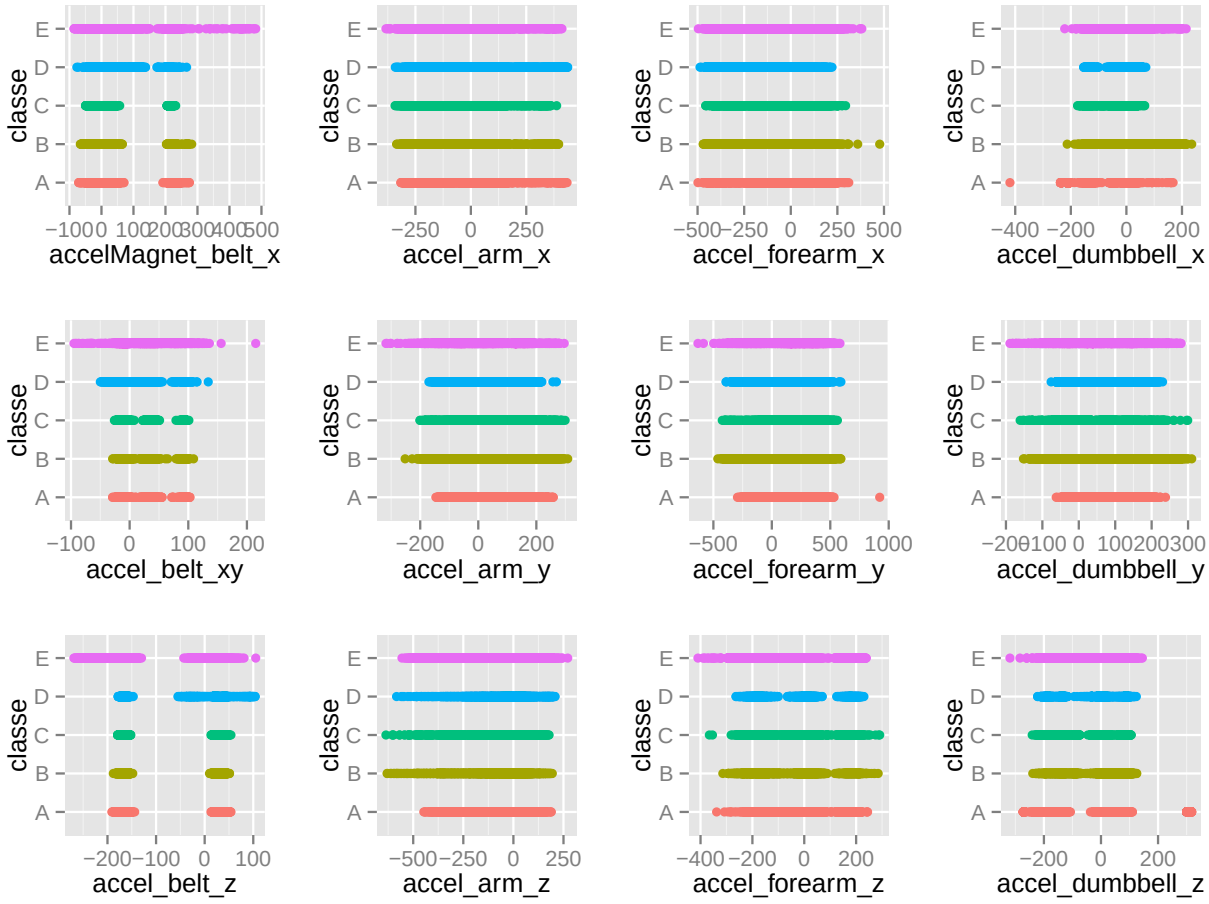
Remove Highly Correlated Variables

Create _xyz Training and Validation Set

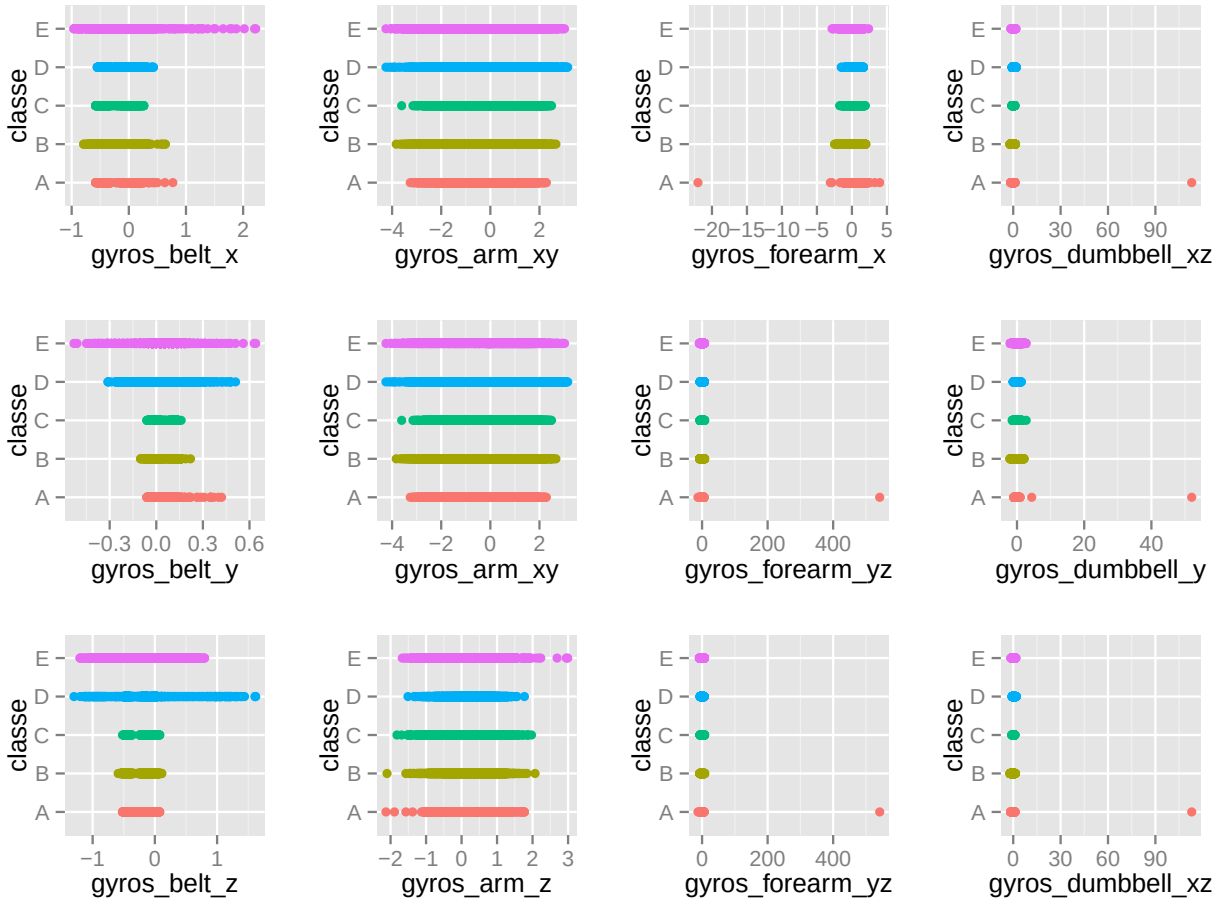
I used the recommended value of .6 to create a training set which leaves 40% for a validation set in order to calculate an out of sample error rate.

```
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

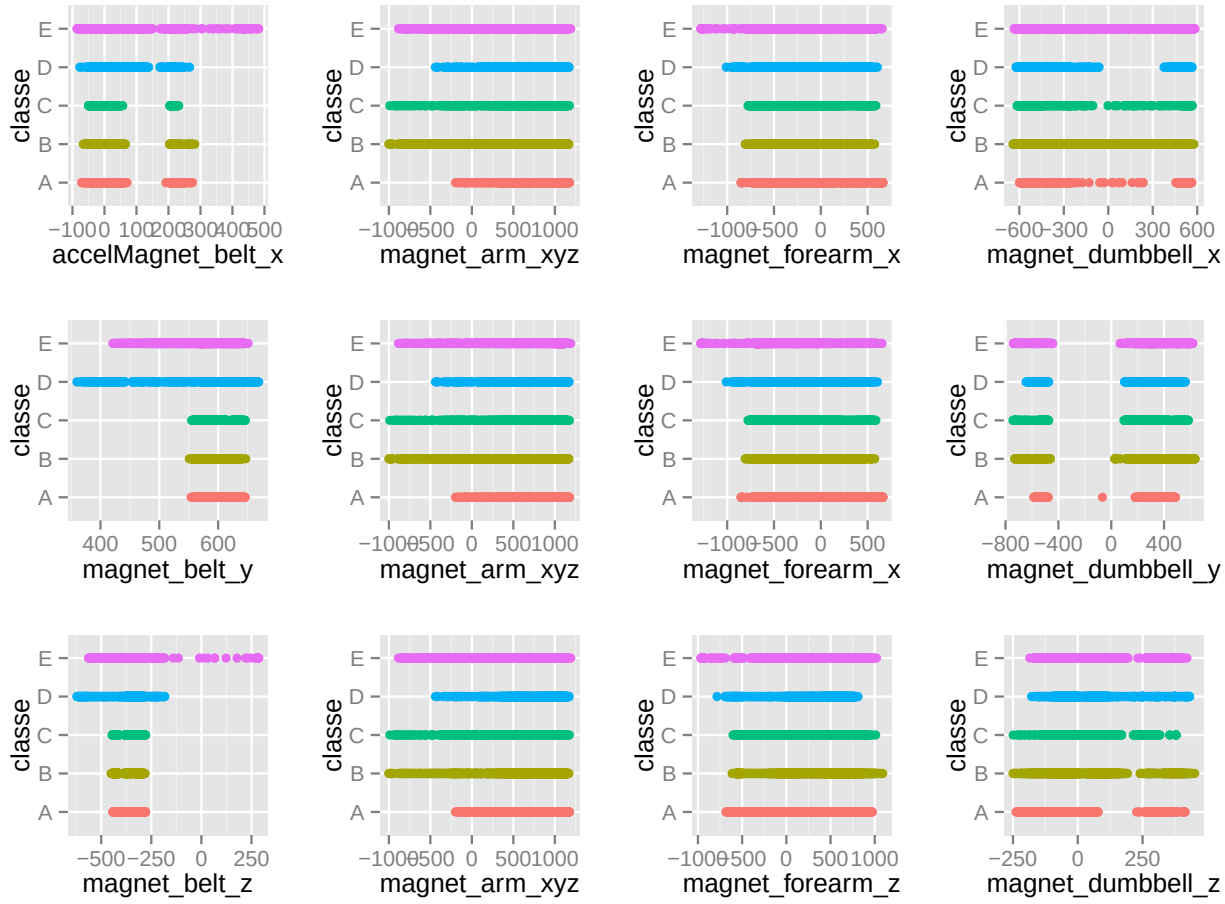


Explore _xyz Gyroscope Variables Graphically



Classe A is the correct lifting method and there appear to be some outliers in forearm and dumbbell measurements. These points may lead to overfitting problems.

Explore _xyz Magnetometer Variables Graphically



I did not find any of the graph panels to be helpful.

Run rpart

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

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

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

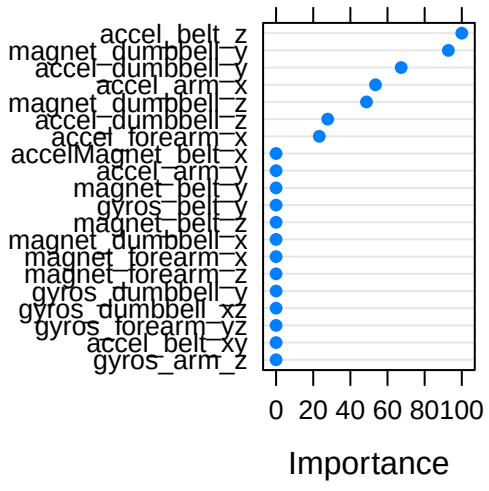
```
xyzFit <- train(classe ~., data = training_xyz,
          method = "rpart",
          control = rpart.control(minsplit = 2),
          trControl = trCtrl,
          tuneLength = 5)
```

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

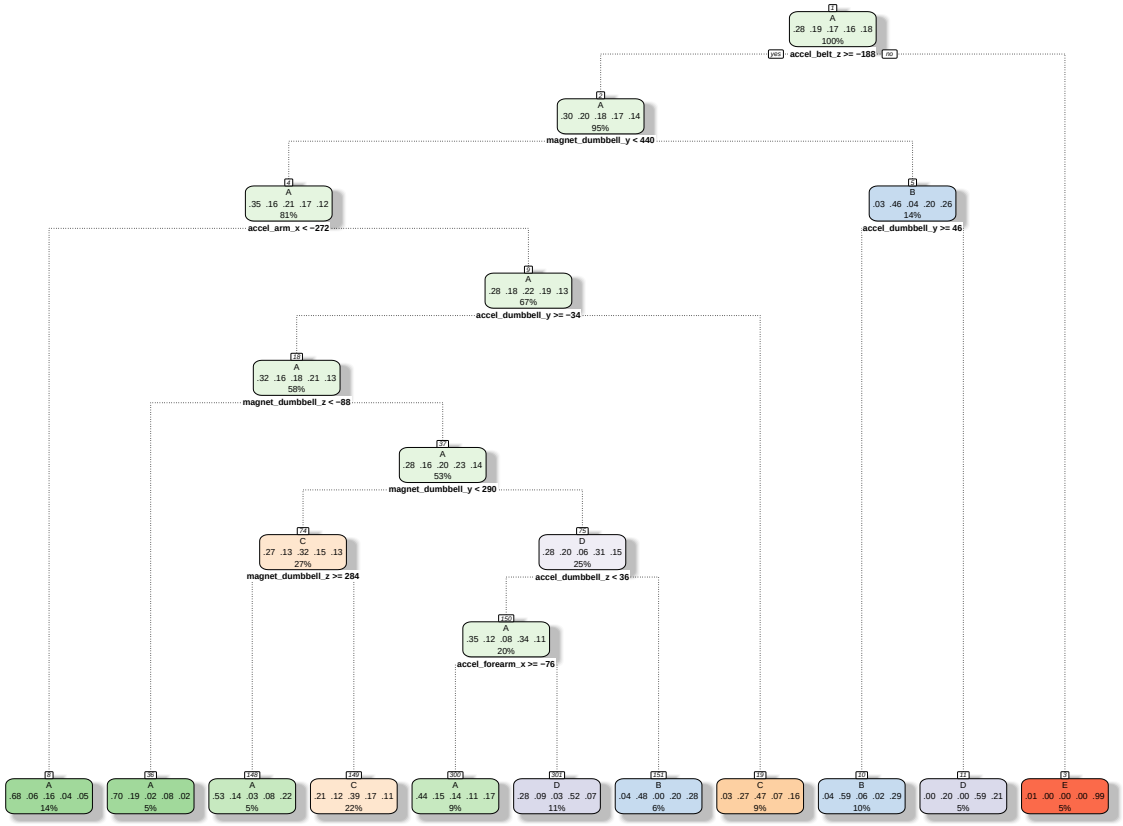
```
varImp(xyzFit, compete= FALSE)
```

```
## 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="")
```




```
xyzFitTrainPred <- predict(xyzFit, newdata = training_xyz)
confusionMatrix(xyzFitTrainPred, training_xyz$classe)
```

Compare Prediction with Training Data Set

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2315  451  435  274  401
##          B   74  983   70  153  506
##          C  588  613 1508  514  446
##          D  366  231   41  989  202
##          E    5    1    0    0  610
##
## Overall Statistics
##
##                Accuracy : 0.5439
```

```
##                  95% CI : (0.5349, 0.5529)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4209
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.6915  0.43133   0.7342  0.51244  0.28176
## Specificity            0.8148  0.91545   0.7777  0.91469  0.99938
## Pos Pred Value         0.5973  0.55039   0.4110  0.54073  0.99026
## Neg Pred Value         0.8692  0.87027   0.9327  0.90540  0.86066
## Prevalence             0.2843  0.19353   0.1744  0.16389  0.18385
## Detection Rate         0.1966  0.08347   0.1281  0.08398  0.05180
## Detection Prevalence   0.3291  0.15166   0.3116  0.15532  0.05231
## Balanced Accuracy      0.7531  0.67339   0.7559  0.71356  0.64057
```

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

Compare Prediction with Valadation Data Set

```
## 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
```

Out of Sample Error The validation set was held out inorder to be a reasonable estimate of how the model would behaving on the testing data. The allows an out of sample error rate to be calculated. Generally the accuracy will be lower on the validation set than it is on the training set, however, my model performs slightly better on the validation data set. The Kappa statistic takes into account the expected error rate and is the (Observed accuracy - Expected accuracy) / (1 - Expected Accuracy)

Training Accurancy	Training Kappa	Validation Accuracy	Validation Kappa
0.5439 (0.5349, 0.5529)	0.4209 0	.557 (0.5459, 0.568) 0	.4376

Load Testing Data Set and Adjust to Match Training Data

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

```
testing <- select(pml_testing, contains("_x"), contains("_y"), contains( "_z"), -contains("stddev"), -c
```

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

```
testing <- select(testing,
              -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)
```

```
testingPred <- predict(xyzFit, newdata = testing)
testingPred
```

Predict Testing Data Set

```
##  [1] C A A C A C D D A A A A A A A B C D C B
## Levels: A B C D E
```