Classify how well an exercise (dumbell curl) was done

```
## Warning: package 'caret' was built under R version 3.2.3
```

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

Why

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. 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. The machine learning algorithm will use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Cleaning the data

The original data contains 160 features for 19622 observations. To clean the data non zero values (columns) and columns with large numbers of missing values were removed.

```
    nzv <- nearZeroVar(pml,saveMetrics=TRUE)
    pml_nzv <- pml[,!nzv[,4]]
    work <- is.na(pml_nzv)
    work_list <- (colSums(work)==max(colSums(work)))
    pml_stripped <- pml_nzv[,!work_list]
    pml_stripped_features <- dim(pml_stripped)[2]
```

This reduces the number of features to 59. The next step was to remove highly correlated variables.

```
    pml_numeric_cor <- pml_stripped[,7:58]
    pml_cor <- abs(cor(pml_numeric_cor))
    diag(pml_cor)<-0
    corTable<-which(pml_cor>.8, arr.ind=TRUE)
    corLabel<-cbind(rownames(pml_cor)[corTable[,1]],colnames(pml_cor)[corTable[,2]])
    highlyCorDescr <- findCorrelation(pml_cor, cutoff = .75)
    pml_filtered <- pml_stripped[,(-(highlyCorDescr+6))]
```

The next step is to remove linear combinations

```
    linearcombos <- findLinearCombos(pml_filtered[,7:37])$remove
```

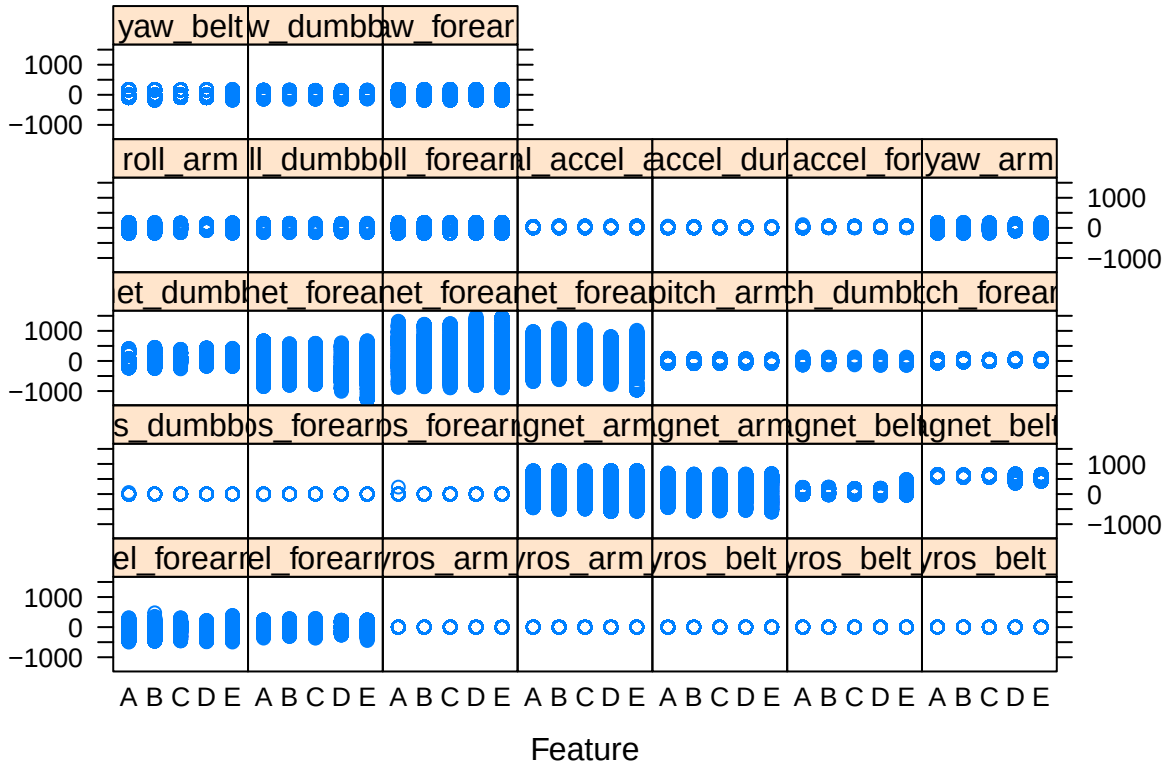
Use principal component analysis to reduce the number of variables

```
    work <- prcomp(pml_filtered[,7:37],center=TRUE, scale. = TRUE)
    pcaOutput <- preProcess(pml_filtered[,7:37], method=c("BoxCox","center","scale","pca"))
    pml_train <- predict(pcaOutput,pml_filtered[,7:38])
```

Plot of features

Based on the following plot all of the features needed to be included as potential candidates to include in the machine learning.

```
featurePlot(pml_filtered[,7:37],pml_filtered$classe)
```



Classification model

The assignment is to create a model that accurately determines which class the observation is based on the information about the movement of sensors attached to the person. A number of methods can be used to accomplish this.

The first method tried was linear discriminate analysis.

```
    pml_lda <- train(pml_train$classe ~ ., method="lda", data=pml_train)
```

```
## Loading required package: MASS
```

```
    pml_lda_result <- predict(pml_lda,pml_train)
    pml_lda_cm <-confusionMatrix(pml_lda_result,pml_train$classe)
```

The next method was Stochastic Gradient Boosting (also known as Gradient Boosted Machine or GBM)

```
    pml_gbm <- train(pml_train$classe ~ ., method="gbm", data=pml_train,verbose=FALSE)
```

```
## Loading required package: gbm

## Warning: package 'gbm' was built under R version 3.2.3

## Loading required package: survival
##
## Attaching package: 'survival'
##
## The following object is masked from 'package:caret':
##
##     cluster
##
## Loading required package: splines
## Loading required package: parallel
## Loaded gbm 2.1.1
## Loading required package: plyr
```

```
    pml_gbm_result <- predict(pml_gbm,pml_train)
    pml_gbm_cm <-confusionMatrix(pml_gbm_result,pml_train$classe)
```

The final method was Random Forest

```
    pml_rf <-train(classe~.,data=pml_train,method="rf",
                trControl=trainControl(method="cv",number=5),
                prox=TRUE,allowParallel=TRUE)
```

```
## Loading required package: randomForest

## Warning: package 'randomForest' was built under R version 3.2.3

## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```
    pml_rf_result <- predict(pml_rf,pml_train)
    pml_rf_cm <-confusionMatrix(pml_rf_result,pml_train$classe)
```

Accuracy of Classification model

Based on the following estimates of model fit, the Random Forest generates the best results.

The accuracy - how often the classifier generated the correct result is

1. Linear Discriminate analysis 0.5216084
2. Gradient Boosted Machine 0.8400265
3. Random Forest 1

The sensitivity - the percentage of observation that the classifier predicts an event will occur correctly

1. Linear Discriminate analysis 0.6713262, 0.3979457, 0.5289305, 0.4446517, 0.4818409
2. Gradient Boosted Machine 0.9265233, 0.7821965, 0.8477499, 0.7991294, 0.7962296
3. Random Forest 1, 1, 1, 1, 1

The specificity - percentage of observations that the classifier predicts an event will not occur correctly

1. Linear Discriminate analysis 0.8369178, 0.8498578, 0.8564815, 0.8980861, 0.9547924
2. Gradient Boosted Machine 0.9549922, 0.9606319, 0.9335185, 0.9650738, 0.9853887
3. Random Forest 1, 1, 1, 1, 1