Practical Machine Project.

Executive Summary: Human Activity Recognition has become a significant research area. 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. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants and predict the manner in which they did the exercise. There are five classes of activity. They are sitting, sitting down, standing, standing up and walking.

Using the cross validation method, multiple models were developed and tested. The model with the smallest error was selected to predict the 20 cases in the pml_tetsing file.

Data Processing Two files have been provided for this project. A training file and a testing file. The testing file will be used for the project submission. The training dataset will be further split into train and test for model development and validation. Utilizing the head and str functions, we'll review the structure of the data.

Many data points have "NA" values and will need to be removed from the data. Additionally, columns with values of #DIV/0! will also need to be handled.

```r
testing_tmp = read.csv("pml-testing.csv", header = TRUE,na.strings=c("","NA"))
training_tmp = read.csv("pml-training.csv", header = TRUE,na.strings=c("","NA"))
str(training_tmp)
```

```
## 'data.frame':    19622 obs. of  160 variables:
##  $ X                       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ user_name               : Factor w/ 6 levels "adelmo","carlitos",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ raw_timestamp_part_1    : int  1323084231 1323084231 1323084231 1323084232 1323084232 1323084232 1
##  $ raw_timestamp_part_2    : int  788290 808298 820366 120339 196328 304277 368296 440390 484323 4844
##  $ cvtd_timestamp          : Factor w/ 20 levels "02/12/2011 13:32",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ new_window              : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ num_window              : int  11 11 11 12 12 12 12 12 12 12 ...
##  $ roll_belt               : num  1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...
##  $ pitch_belt              : num  8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...
##  $ yaw_belt                : num  -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...
##  $ total_accel_belt        : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ kurtosis_roll_belt      : Factor w/ 396 levels "-0.016850","-0.021024",..: NA NA NA NA NA NA NA NA
##  $ kurtosis_picth_belt     : Factor w/ 316 levels "-0.021887","-0.060755",..: NA NA NA NA NA NA NA NA
##  $ kurtosis_yaw_belt       : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_belt      : Factor w/ 394 levels "-0.003095","-0.010002",..: NA NA NA NA NA NA NA NA
##  $ skewness_roll_belt.1    : Factor w/ 337 levels "-0.005928","-0.005960",..: NA NA NA NA NA NA NA NA
##  $ skewness_yaw_belt       : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ max_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_belt            : Factor w/ 67 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_belt            : Factor w/ 67 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_belt    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_belt      : Factor w/ 3 levels "#DIV/0!","0.00",..: NA NA NA NA NA NA NA NA NA NA .
##  $ var_total_accel_belt    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_belt        : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ var_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_belt       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_belt         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_belt_x            : num  0 0.02 0 0.02 0.02 0.02 0.02 0.02 0.02 0.03 ...
##  $ gyros_belt_y            : num  0 0 0 0 0.02 0 0 0 0 0 ...
##  $ gyros_belt_z            : num  -0.02 -0.02 -0.02 -0.03 -0.02 -0.02 -0.02 -0.02 -0.02 0 ...
##  $ accel_belt_x            : int  -21 -22 -20 -22 -21 -21 -22 -22 -20 -21 ...
##  $ accel_belt_y            : int  4 4 5 3 2 4 3 4 2 4 ...
##  $ accel_belt_z            : int  22 22 23 21 24 21 21 21 24 22 ...
##  $ magnet_belt_x           : int  -3 -7 -2 -6 -6 0 -4 -2 1 -3 ...
##  $ magnet_belt_y           : int  599 608 600 604 600 603 599 603 602 609 ...
##  $ magnet_belt_z           : int  -313 -311 -305 -310 -302 -312 -311 -313 -312 -308 ...
##  $ roll_arm                : num  -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...
##  $ pitch_arm               : num  22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...
##  $ yaw_arm                 : num  -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...
##  $ total_accel_arm         : int  34 34 34 34 34 34 34 34 34 34 ...
##  $ var_accel_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_arm         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_arm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_arm_x             : num  0 0.02 0.02 0.02 0 0.02 0 0.02 0.02 0.02 ...
##  $ gyros_arm_y             : num  0 -0.02 -0.02 -0.03 -0.03 -0.03 -0.03 -0.02 -0.03 -0.03 ...
##  $ gyros_arm_z             : num  -0.02 -0.02 -0.02 0.02 0 0 0 0 -0.02 -0.02 ...
##  $ accel_arm_x             : int  -288 -290 -289 -289 -289 -289 -289 -289 -288 -288 ...
##  $ accel_arm_y             : int  109 110 110 111 111 111 111 111 109 110 ...
##  $ accel_arm_z             : int  -123 -125 -126 -123 -123 -122 -125 -124 -122 -124 ...
##  $ magnet_arm_x            : int  -368 -369 -368 -372 -374 -369 -373 -372 -369 -376 ...
##  $ magnet_arm_y            : int  337 337 344 344 337 342 336 338 341 334 ...
##  $ magnet_arm_z            : int  516 513 513 512 506 513 509 510 518 516 ...
##  $ kurtosis_roll_arm       : Factor w/ 329 levels "-0.02438","-0.04190",..: NA NA NA NA NA NA NA NA N
##  $ kurtosis_picth_arm      : Factor w/ 327 levels "-0.00484","-0.01311",..: NA NA NA NA NA NA NA NA N
##  $ kurtosis_yaw_arm        : Factor w/ 394 levels "-0.01548","-0.01749",..: NA NA NA NA NA NA NA NA N
##  $ skewness_roll_arm       : Factor w/ 330 levels "-0.00051","-0.00696",..: NA NA NA NA NA NA NA NA N
##  $ skewness_pitch_arm      : Factor w/ 327 levels "-0.00184","-0.01185",..: NA NA NA NA NA NA NA NA N
##  $ skewness_yaw_arm        : Factor w/ 394 levels "-0.00311","-0.00562",..: NA NA NA NA NA NA NA NA N
##  $ max_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_arm       : int  NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ roll_dumbbell           : num  13.1 13.1 12.9 13.4 13.4 ...
##  $ pitch_dumbbell          : num  -70.5 -70.6 -70.3 -70.4 -70.4 ...
##  $ yaw_dumbbell            : num  -84.9 -84.7 -85.1 -84.9 -84.9 ...
##  $ kurtosis_roll_dumbbell  : Factor w/ 397 levels "-0.0035","-0.0073",..: NA NA NA NA NA NA NA NA NA
##  $ kurtosis_picth_dumbbell : Factor w/ 400 levels "-0.0163","-0.0233",..: NA NA NA NA NA NA NA NA NA
##  $ kurtosis_yaw_dumbbell   : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_dumbbell  : Factor w/ 400 levels "-0.0082","-0.0096",..: NA NA NA NA NA NA NA NA NA
##  $ skewness_pitch_dumbbell : Factor w/ 401 levels "-0.0053","-0.0084",..: NA NA NA NA NA NA NA NA NA
##  $ skewness_yaw_dumbbell   : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ max_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_dumbbell        : Factor w/ 72 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_dumbbell        : Factor w/ 72 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##   [list output truncated]
```

Review the values and frequency of the dependent 'classe' variable.

```r
table(training_tmp$classe)
```

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

Remove NA's and keep only data elements used for modeling.

```r
training_tmp1 <- na.omit(training_tmp)

training_tmp1 <- training_tmp1[,c(8:11,37:49,60:68,84:86,113:124,140,151:160)]
summary(training_tmp1)
```

```
##    roll_belt        pitch_belt         yaw_belt       total_accel_belt
##  Min.   :-27.80   Min.   :-51.60   Min.   :-175.00   Min.   : 1.0
##  1st Qu.:  1.15   1st Qu.:  1.41   1st Qu.: -88.20   1st Qu.: 4.0
##  Median :116.00   Median :  5.38   Median :  -6.70   Median :17.0
##  Mean   : 68.16   Mean   :  0.48   Mean   :  -8.52   Mean   :11.8
##  3rd Qu.:123.00   3rd Qu.: 15.53   3rd Qu.:  14.50   3rd Qu.:18.0
##  Max.   :161.00   Max.   : 60.00   Max.   : 177.00   Max.   :27.0
##   gyros_belt_x     gyros_belt_y      gyros_belt_z     accel_belt_x
##  Min.   :-0.740   Min.   :-0.2200   Min.   :-1.030   Min.   :-66.0
##  1st Qu.:-0.045   1st Qu.: 0.0000   1st Qu.:-0.200   1st Qu.:-21.0
##  Median : 0.030   Median : 0.0200   Median :-0.120   Median :-15.0
##  Mean   :-0.005   Mean   : 0.0446   Mean   :-0.128   Mean   : -5.5
##  3rd Qu.: 0.110   3rd Qu.: 0.1100   3rd Qu.: 0.000   3rd Qu.: -5.0
##  Max.   : 0.670   Max.   : 0.4200   Max.   : 1.280   Max.   : 72.0
##   accel_belt_y    accel_belt_z  magnet_belt_x    magnet_belt_y
##  Min.   :-18.0   Min.   :-248   Min.   :-24.0   Min.   :388
##  1st Qu.:  3.0   1st Qu.:-162   1st Qu.: 12.0   1st Qu.:581
##  Median : 42.0   Median :-154   Median : 36.0   Median :601
##  Mean   : 32.1   Mean   : -78   Mean   : 57.0   Mean   :593
```

```
##  3rd Qu.: 62.0   3rd Qu.:  28   3rd Qu.: 60.8   3rd Qu.:611
##  Max.   : 79.0   Max.   :  92   Max.   :383.0   Max.   :655
##  magnet_belt_z     roll_arm        pitch_arm        yaw_arm
##  Min.   :-604   Min.   :-176.0   Min.   :-79.5   Min.   :-175.0
##  1st Qu.:-372   1st Qu.: -35.0   1st Qu.:-22.9   1st Qu.: -43.5
##  Median :-322   Median :   0.0   Median :  0.0   Median :   0.0
##  Mean   :-346   Mean   :  14.8   Mean   : -3.5   Mean   :  -0.2
##  3rd Qu.:-306   3rd Qu.:  74.5   3rd Qu.: 11.8   3rd Qu.:  46.1
##  Max.   : 289   Max.   : 167.0   Max.   : 79.8   Max.   : 157.0
##  total_accel_arm  gyros_arm_x      gyros_arm_y      gyros_arm_z
##  Min.   : 3.0    Min.   :-5.220   Min.   :-3.440   Min.   :-1.440
##  1st Qu.:17.0    1st Qu.:-1.488   1st Qu.:-0.900   1st Qu.:-0.070
##  Median :27.0    Median : 0.080   Median :-0.220   Median : 0.260
##  Mean   :25.2    Mean   : 0.033   Mean   :-0.246   Mean   : 0.274
##  3rd Qu.:33.0    3rd Qu.: 1.692   3rd Qu.: 0.155   3rd Qu.: 0.710
##  Max.   :59.0    Max.   : 4.420   Max.   : 2.380   Max.   : 1.620
##   accel_arm_x      accel_arm_y      accel_arm_z      magnet_arm_x
##  Min.   :-307.0   Min.   :-245.0   Min.   :-531.0   Min.   :-539
##  1st Qu.:-235.0   1st Qu.: -51.8   1st Qu.:-132.0   1st Qu.:-258
##  Median : -51.5   Median :   4.5   Median : -34.0   Median : 289
##  Mean   : -59.3   Mean   :  30.3   Mean   : -63.7   Mean   : 199
##  3rd Qu.:  78.0   3rd Qu.: 136.8   3rd Qu.:  34.0   3rd Qu.: 602
##  Max.   : 435.0   Max.   : 261.0   Max.   : 199.0   Max.   : 773
##   magnet_arm_y     magnet_arm_z  roll_dumbbell    pitch_dumbbell
##  Min.   :-392.0   Min.   :-584   Min.   :-151.8   Min.   :-103.9
##  1st Qu.:   7.2   1st Qu.: 177   1st Qu.: -20.6   1st Qu.: -42.2
##  Median : 216.5   Median : 460   Median :  47.4   Median : -21.9
##  Mean   : 163.1   Mean   : 318   Mean   :  21.3   Mean   : -12.3
##  3rd Qu.: 318.0   3rd Qu.: 543   3rd Qu.:  67.5   3rd Qu.:  16.8
##  Max.   : 466.0   Max.   : 678   Max.   : 151.0   Max.   : 101.7
##   yaw_dumbbell     gyros_dumbbell_x gyros_dumbbell_y  gyros_dumbbell_z
##  Min.   :-129.5   Min.   :-1.540   Min.   :-1.8800   Min.   :-1.820
##  1st Qu.: -76.6   1st Qu.:-0.020   1st Qu.:-0.1600   1st Qu.:-0.330
##  Median : -13.3   Median : 0.140   Median : 0.0200   Median :-0.150
##  Mean   :  -1.0   Mean   : 0.188   Mean   : 0.0453   Mean   :-0.154
##  3rd Qu.:  75.2   3rd Qu.: 0.370   3rd Qu.: 0.2100   3rd Qu.: 0.020
##  Max.   : 150.1   Max.   : 1.830   Max.   : 2.7300   Max.   : 1.310
##  accel_dumbbell_x  accel_dumbbell_y accel_dumbbell_z magnet_dumbbell_x
##  Min.   :-235.00   Min.   :-148.0   Min.   :-271.0   Min.   :-627
##  1st Qu.: -53.75   1st Qu.: -10.0   1st Qu.:-145.0   1st Qu.:-537
##  Median : -17.00   Median :  44.5   Median :  -4.0   Median :-485
##  Mean   : -33.24   Mean   :  53.1   Mean   : -41.8   Mean   :-330
##  3rd Qu.:   7.75   3rd Qu.: 116.0   3rd Qu.:  35.0   3rd Qu.:-297
##  Max.   : 212.00   Max.   : 281.0   Max.   : 317.0   Max.   : 592
##  magnet_dumbbell_y magnet_dumbbell_z  roll_forearm    pitch_forearm
##  Min.   :-723      Min.   :-244.0    Min.   :-180.0   Min.   :-72.5
##  1st Qu.: 233      1st Qu.: -46.0    1st Qu.: -19.4   1st Qu.:  0.0
##  Median : 312      Median :  11.5    Median :  12.8   Median : 12.3
##  Mean   : 220      Mean   :  39.1    Mean   :  31.4   Mean   : 12.4
##  3rd Qu.: 391      3rd Qu.:  93.8    3rd Qu.: 140.0   3rd Qu.: 29.0
##  Max.   : 631      Max.   : 421.0    Max.   : 180.0   Max.   : 83.5
##   yaw_forearm   total_accel_forearm gyros_forearm_x  gyros_forearm_y
##  Min.   :-179   Min.   : 1.0        Min.   :-3.080   Min.   :-5.490
##  1st Qu.: -76   1st Qu.:29.0        1st Qu.:-0.240   1st Qu.:-1.408
```

```
##  Median :   0   Median :34.0        Median : 0.030   Median : 0.050
##  Mean   :  13   Mean   :34.3        Mean   : 0.139   Mean   : 0.077
##  3rd Qu.: 107   3rd Qu.:41.0        3rd Qu.: 0.660   3rd Qu.: 1.550
##  Max.   : 176   Max.   :62.0        Max.   : 1.560   Max.   : 5.640
##  gyros_forearm_z  accel_forearm_x  accel_forearm_y  accel_forearm_z
##  Min.   :-7.940   Min.   :-464.0   Min.   :-406.0   Min.   :-329.0
##  1st Qu.:-0.175   1st Qu.:-187.5   1st Qu.:  39.5   1st Qu.:-181.0
##  Median : 0.075   Median : -64.0   Median : 162.0   Median : -37.5
##  Mean   : 0.116   Mean   : -76.6   Mean   : 152.1   Mean   : -58.7
##  3rd Qu.: 0.480   3rd Qu.:  50.8   3rd Qu.: 304.8   3rd Qu.:  20.0
##  Max.   : 2.230   Max.   : 276.0   Max.   : 583.0   Max.   : 254.0
##  magnet_forearm_x  magnet_forearm_y magnet_forearm_z classe
##  Min.   :-1270.0   Min.   :-835.0   Min.   :-955     A:109
##  1st Qu.: -617.5   1st Qu.: -59.8   1st Qu.: 240     B: 79
##  Median : -416.5   Median : 573.0   Median : 512     C: 70
##  Mean   : -327.3   Mean   : 355.8   Mean   : 403     D: 69
##  3rd Qu.:  -97.2   3rd Qu.: 724.8   3rd Qu.: 658     E: 79
##  Max.   :  655.0   Max.   :1450.0   Max.   : 953
```

Model Summary

Three models with differing splits between train and test have been developed and scored. Of the three models, the best performing model based on the estimated error from cross-validation is the last model developed.

This model was developed with a 80/20 split between train and test. The estimated error is 26% which is the least of the three models.

Create the initial training and testing modeling datasets. The split between the training and testing datasets will be 70/30.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.1.2
```

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

```
inTrain1 <- createDataPartition(y=training_tmp1$classe,
                               p=0.7,list=FALSE)
training <- training_tmp1[inTrain1,]
testing <- training_tmp1[-inTrain1,]
dim(training); dim(testing)
```

```
## [1] 287  52
```

```
## [1] 119  52
```

Create, score and cross validate the initial model.

```r
library(caret)
modFit <- train(classe ~ .,data=training,method="rf",prox=TRUE)
```

```
## Loading required package: randomForest

## Warning: package 'randomForest' was built under R version 3.1.2

## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```r
modFit
```

```
## Random Forest
##
## 287 samples
##  51 predictor
##   5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 287, 287, 287, 287, 287, 287, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa   Accuracy SD  Kappa SD
##    2    0.6465    0.5507  0.03841      0.04904
##   26    0.6614    0.5716  0.04908      0.06116
##   51    0.6436    0.5491  0.04792      0.05995
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 26.
```

```r
pred <- predict(modFit,testing);testing$predRight <- pred==testing$classe
table(pred,testing$classe)
```

```
##
## pred  A  B  C  D  E
##    A 22  1  1  4  0
##    B  6 16  3  2  1
##    C  0  4 17  2  7
##    D  1  1  0 11  2
##    E  3  1  0  1 13
```

```r
print(modFit$finalModel)
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry, proximity = TRUE)
##                Type of random forest: classification
```

```
##                      Number of trees: 500
## No. of variables tried at each split: 26
##
##         OOB estimate of  error rate: 26.13%
## Confusion matrix:
##    A  B  C  D  E class.error
## A 67  3  4  1  2      0.1299
## B  8 33  9  5  1      0.4107
## C  5  2 37  3  2      0.2449
## D  6  4  7 31  1      0.3673
## E  1  5  4  2 44      0.2143
```

Create, score and cross validate the second version of the model. The split between training and testing is 50/50.

```r
library(caret)
inTrain2 <- createDataPartition(y=training_tmp1$classe,
                               p=0.5,list=FALSE)
training2 <- training_tmp1[inTrain2,]
testing2 <- training_tmp1[-inTrain2,]
dim(training2); dim(testing2)
```

```
## [1] 205  52
```

```
## [1] 201  52
```

```r
library(caret)
modFit2 <- train(classe ~ .,data=training2,method="rf",prox=TRUE)
modFit2
```

```
## Random Forest
##
## 205 samples
##  51 predictor
##   5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 205, 205, 205, 205, 205, 205, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa   Accuracy SD  Kappa SD
##    2    0.5717    0.4543  0.05552      0.06985
##   26    0.5785    0.4649  0.05946      0.07487
##   51    0.5590    0.4400  0.06309      0.07927
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 26.
```

```r
pred <- predict(modFit,testing2);testing2$predRight <- pred==testing2$classe
table(pred,testing2$classe)
```

```
##
## pred  A  B  C  D  E
##    A 50  0  1  2  0
##    B  2 35  2  2  1
##    C  0  2 32  2  4
##    D  1  1  0 27  1
##    E  1  1  0  1 33
```

```r
print(modFit2$finalModel)
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry, proximity = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 26
##
##         OOB estimate of  error rate: 34.63%
## Confusion matrix:
##    A  B  C  D  E class.error
## A 47  4  1  0  3      0.1455
## B  7 21  5  7  0      0.4750
## C  8  4 22  1  0      0.3714
## D  7  5  3 18  2      0.4857
## E  2  3  5  4 26      0.3500
```

Create the final training and testing modeling datasets. The split between the training and testing datasets will be 80/20.

```r
library(caret)
inTrain3 <- createDataPartition(y=training_tmp1$classe,
                               p=0.8,list=FALSE)
training3 <- training_tmp1[inTrain3,]
testing3 <- training_tmp1[-inTrain3,]
dim(training3); dim(testing3)
```

```
## [1] 328  52
```

```
## [1] 78 52
```

```r
library(caret)
modFit3 <- train(classe ~ .,data=training3,method="rf",prox=TRUE)
modFit3
```

```
## Random Forest
##
## 328 samples
```

```
##  51 predictor
##   5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 328, 328, 328, 328, 328, 328, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa   Accuracy SD  Kappa SD
##    2    0.6811    0.5965  0.03994      0.05128
##   26    0.6978    0.6187  0.04025      0.05047
##   51    0.6822    0.5989  0.04148      0.05109
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 26.
```

```r
pred <- predict(modFit,testing3);testing3$predRight <- pred==testing3$classe
table(pred,testing3$classe)
```

```
##
## pred  A  B  C  D  E
##    A 20  0  0  1  0
##    B  1 15  0  1  0
##    C  0  0 14  0  1
##    D  0  0  0 11  0
##    E  0  0  0  0 14
```

```r
print(modFit3$finalModel)
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry, proximity = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 26
##
##         OOB estimate of  error rate: 23.17%
## Confusion matrix:
##    A  B  C  D  E class.error
## A 77  2  4  4  1      0.1250
## B  4 45 10  5  0      0.2969
## C  7  2 44  3  0      0.2143
## D  1  3  9 41  2      0.2679
## E  2  5  6  6 45      0.2969
```