Human Activity Recognition

Aim of the project

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Different outcomes

1. Class A - Exactly according to the specification.
2. Class B - Throwing the elbows to the front.
3. Class C - Lifting the dumbbell only halfway.
4. Class D - Lowering the dumbbell only halfway.
5. Class E - Throwing the hips to the front.

Downloading data

```
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", destfile = "train
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", destfile = "test.cs
```

Importing required libraries

```
library(caret)
```

Preprocessing

```
training <- read.csv("train.csv", header = T)
testing <- read.csv("test.csv", header = T)

dim(training)
```

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

```
##Lets see how many columns have all data as NA values
columnNAs <- colSums(is.na(training)) == 19216
training[, columnNAs] <- NULL

## Now the dataset has been reduced to 93 columns.
dim(training)
```

```
## [1] 19622    93
```

```
##Applying same for the test dataset
testing[, columnNAs] <- NULL
dim(testing)
```

```
## [1] 20 93
```

```
##Lets filter the required columns
requiredColumns1 <- grep(".*arm", colnames(training))
requiredColumns2 <- grep(".*belt", colnames(training))
requiredColumns3 <- grep(".*dumbbell", colnames(training))

## These are the columns that are required for our prediction.
requiredColumns1; requiredColumns2; requiredColumns3
```

```
##  [1] 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 71 72 73 74 75 76
## [26] 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92
```

```
##  [1]  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
```

```
##  [1] 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70
```

```
## We can drop off the columns which have not much variance in them.
nzv <- nearZeroVar(training, saveMetrics = TRUE)
training <- training[, (nzv$nzv == FALSE)]

dim(training)
```

```
## [1] 19622    59
```

Model Selection

We have filtered out what are the columns to be useful for prediction. By reading the dataset information from the website, one can easily determine that those columns which contains accelerometer are very important that cannot be eliminated.

Let's consider a simple Decision Treee and see how much accuracy we get.

```
set.seed(1223)
validation <- createDataPartition(y = training$classe, p = 0.25, list = F)
training_data <- training[-validation,]
validation_data <- training[validation,]

modelDT <- train(classe ~ accel_belt_x + accel_belt_y + accel_belt_z + accel_arm_x + accel_arm_y + accel
                 accel_dumbbell_x + accel_dumbbell_y+ accel_dumbbell_z +
                 accel_forearm_x + accel_forearm_y + accel_forearm_z,
                 data = training_data, method = "rpart")
modelDT
```

```
## CART
##
## 14715 samples
##    12 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 14715, 14715, 14715, 14715, 14715, 14715, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.02445394  0.4480776  0.28615746
##   0.03950617  0.3800915  0.16479292
##   0.07302944  0.3209736  0.05521437
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.02445394.
```

Here, we get only 44 percent accuracy which defines that this is a bad model to work with. So, we should go for random forest.

Random Forest

Initially, only accelerometer columns where selected and trained for **Random Forest**. But the accuracy was only 94 percent. In order to improve the accuracy, we should find some other variables which are all important for the prediction.

```
set.seed(1223)
validation <- createDataPartition(y = training$classe, p = 0.25, list = F)
training_data <- training[-validation,]
validation_data <- training[validation,]


modelFit <- train(classe ~ accel_belt_x + accel_belt_y + accel_belt_z + accel_arm_x + accel_arm_y + acc
                  accel_dumbbell_x + accel_dumbbell_y+ accel_dumbbell_z +
                  accel_forearm_x + accel_forearm_y + accel_forearm_z +
                  roll_belt + pitch_belt + yaw_belt +
                  roll_arm + pitch_arm + yaw_arm +
                  roll_forearm + pitch_forearm + yaw_forearm +
                  roll_dumbbell + pitch_dumbbell + yaw_dumbbell,
                  data = training_data, method = "rf")
modelFit
```

```
## Random Forest
##
## 14715 samples
##    24 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 14715, 14715, 14715, 14715, 14715, 14715, ...
```

```
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9859441  0.9822133
##   13    0.9857536  0.9819738
##   24    0.9800056  0.9747012
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

Using random forest, we have got 98 percent accuracy which is quite satisfactory. The next step is to see how well it performs in validation set.

```
predictions <- predict(modelFit, validation_data)
confusionMatrix(validation_data$classe, predictions)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1388    0    5    2    0
##          B    5  944    1    0    0
##          C    0   11  841    4    0
##          D    0    0    8  795    1
##          E    0    0    2    2  898
##
## Overall Statistics
##
##                Accuracy : 0.9916
##                  95% CI : (0.9887, 0.994)
##     No Information Rate : 0.2839
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9894
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9964   0.9885   0.9813   0.9900   0.9989
## Specificity            0.9980   0.9985   0.9963   0.9978   0.9990
## Pos Pred Value         0.9950   0.9937   0.9825   0.9888   0.9956
## Neg Pred Value         0.9986   0.9972   0.9961   0.9981   0.9998
## Prevalence             0.2839   0.1946   0.1746   0.1636   0.1832
## Detection Rate         0.2829   0.1924   0.1714   0.1620   0.1830
## Detection Prevalence   0.2843   0.1936   0.1744   0.1638   0.1838
## Balanced Accuracy      0.9972   0.9935   0.9888   0.9939   0.9989
```

We got 99 percent accuracy for our validation set. The expected out of sample error is approximately 0.01(i.e. 1 - 0,9923).

```r
## Prediction for the 20 new observations.
predict(modelFit, testing)
```

```
##  [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
```

Reference

1. Human Activity Recognition