Prediction Assignment - Human Activity Recognition for Weight Lifting Exercise

Synopsis

This particular document is aimed at predicting how well an activity was performed by a particular participant in a weight lifting exercise.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. In this document I have used 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.The data is available at Weight Lifting Excerice Data I have used Machine Learning algorithms in R to predict the class of WLE dataset. I have first pre processed the data to remove unwanted variables and than used different algorithms on train data to see model accuracy and selected the best among them. Cross Validation set was used for finding out of sample error. Finally I ran my model fit on the testing data provided to check the accuracy.

Data Processing and Cleaning

In this section we read the raw data from the source we obtained, summarize the data and finally transform and clean the data for our prediction based on which we derive our results and make conclusion.

As a first step here, I am cleaning the data to remove all columns having NA values. This gives me 59 predictor variables for prediction variable “classe”. Post that I am removing the variables that I don't think will help much in prediction. Since this is sensor data related prediction we discard the data that we think wont have impact on sensor measure, under certain assumptions. The first 7 variables “X”, “user_name”, “raw_timestamp_part_1”, “raw_timestamp_part_2”, “cvtd_timestamp”, “new_window”, “num_window” are omitted. They don't appear to be related with sensor reading and are mostly the characteristics of the individual undergoing weight lifting exercise. Hence removing them makes sense.

```
## Setting the working directory.
setwd("C:\\Users\\bimehta\\Desktop\\R_Prog\\Machine Learning\\Assignment")
## Reading the Raw Data and making blank values NA
trainingDataRaw <- read.csv("pml-training.csv", header = TRUE, na.strings = c("NA",
    ""))
testingDataRaw <- read.csv("pml-testing.csv", header = TRUE, na.strings = c("NA",
    ""))
## Removing columns with at least one NA value
cleanTrainingData <- trainingDataRaw[, colSums(is.na(trainingDataRaw)) == 0]
finalTrainingData <- cleanTrainingData[, 8:60]
cleanTestingData <- testingDataRaw[, colSums(is.na(testingDataRaw)) == 0]
finalTestingData <- cleanTestingData[, 8:60]
```

Summarizing Data

Here we see what is the size of data and what the data looks like and what is the structure of data

```
## Finding Number of Variables and Observations
dim(finalTrainingData)
```

```
## [1] 19622    53
```

Creating Training and Cross Validation Sets

Next I have done is created 2 subsets of data based on “classe”. One is training data set which will be used to create the model fit and the other is cross validation data set which will be used to calculate Out of Sample Error.

```
library(caret)
```

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

```
inBuild <- createDataPartition(y = finalTrainingData$classe, p = 0.7, list = FALSE)
validation <- finalTrainingData[-inBuild, ]
```

```
training <- finalTrainingData[inBuild, ]
dim(training)
```

```
## [1] 13737    53
```

```
dim(validation)
```

```
## [1] 5885   53
```

Model Building

I used two different algorithms here to see which one gives better accuracy. The first algorithm I used is Random Forest. The 2nd model I used was using the Boosting method. Based on the accuracy of the 2 models I have selected the one with higher accuracy and calculated Out of Sample Error in Results section

```
## Creating RF mdoel, predicting and calculating Confusion Matrix on Cross
## Validaton Set
set.seed(6677)
modFitRF <- train(classe ~ ., method = "rf", data = training, trControl = trainControl(method
= "cv",
    number = 4))
```

```
## Loading required package: randomForest
## randomForest 4.6-7
## Type rfNews() to see new features/changes/bug fixes.
```

```
modFitRF
```

```
## Random Forest
##
## 13737 samples
##    52 predictors
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold)
##
## Summary of sample sizes: 10304, 10302, 10303, 10302
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
##   2     1         1      0.002        0.002
##   30    1         1      0.002        0.002
##   50    1         1      0.003        0.004
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

```
validateRF <- predict(modFitRF, newdata = validation)
crf <- confusionMatrix(validateRF, validation$classe)

## Creating GBM model, predicting and calculating Confusion Matrix on Cross
## Validaton Set
set.seed(6688)
modFitGBM <- train(classe ~ ., method = "gbm", data = training, trControl =
trainControl(method = "cv",
    number = 4), verbose = FALSE)
```

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

```
## Loaded gbm 2.1
## Loading required package: plyr
```

```
modFitGBM
```

```
## Stochastic Gradient Boosting
##
## 13737 samples
##    52 predictors
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (4 fold)
##
## Summary of sample sizes: 10304, 10302, 10302, 10303
##
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy  Kappa  Accuracy SD  Kappa SD
##   1                  50       0.8       0.7    0.009        0.01
##   1                  100      0.8       0.8    0.008        0.01
##   1                  200      0.9       0.8    0.006        0.008
##   2                  50       0.9       0.8    0.004        0.005
##   2                  100      0.9       0.9    0.005        0.007
##   2                  200      0.9       0.9    0.006        0.007
##   3                  50       0.9       0.9    0.002        0.002
##   3                  100      0.9       0.9    0.004        0.004
##   3                  200      1         0.9    0.002        0.003
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were n.trees = 150,
##  interaction.depth = 3 and shrinkage = 0.1.
```

```
validateGBM <- predict(modFitGBM, newdata = validation)
cgbm <- confusionMatrix(validateGBM, validation$classe)
```

Results

The accuracy for 2 models on Cross Validation set is shown below:

```
## Random Forest Accuracy on Cross Validation Set
crf$overall[1]
```

```
## Accuracy
##   0.9947
```

```
## Boosting Model Accuracy
cgbm$overall[1]
```

```
## Accuracy
##   0.9638
```

Based on the above we see that Random Forest model is a better fit for the given data. The out of sample error for Cross Validation Set is as below. As we see Out of Sample error is very small and is less than 1% indicating a very good fit.

```
(1 - (sum(validateRF == validation$classe)/length(validateRF))) * 100
```

```
## [1] 0.5268
```

Test Set Prediction

Based on the RF output, I have generated the Prediction on the Test Set and saved it in directory as shown below for submission.

```
testRF <- predict(modFitRF, newdata = finalTestingData)
pml_write_files = function(x) {
```

```
    n = length(x)
    for (i in 1:n) {
        filename = paste0("problem_id_", i, ".txt")
        write.table(x[i], file = filename, quote = FALSE, row.names = FALSE,
            col.names = FALSE)
    }
}
pml_write_files(testRF)
```

Conclusion

Based on the above analysis we were able to obtain a very good predictor model using Random Forest Algorithm. The model worked well on Cross Validation set giving high accuracy and small out of sample error. Using that model we were able to predict the values on our Test Set. If we look at the output of Random Forest model we can see that only 2 variables are good enough predictors to give us high accuracy. It would be worthwhile to find those 2 predictors and build a model and see whether its true or not. There is a good scope of further exploration here.