Practical Machine Learning - Final Project

Project Summary

Using machine learning algorithms, this project will use two data sets for training and testing to predict the manner how individuals did the exercise.

Loading the Libraries and Dataset

After loading the libraries, we will now load the training and testing data sets.The testing dataset will be used later to quiz the model that we have built.

```
library(rpart)
library(rpart.plot)
library(caret)
library(randomForest)

train_data <- read.csv("pml-training.csv", na.strings = c("NA",""))
test_data <- read.csv("pml-testing.csv",na.strings = c("NA",""))
```

Next, we will split the original training data set will be split to 70-30 test-train data.

```
training_partition <- createDataPartition( y = train_data$classe,
                                   p = 0.7,
                                   list = FALSE)
train_prelim <- train_data[training_partition,]
test_prelim <-  train_data[-training_partition,]
```

Data Preprocessing

We will now prepare our data for modeling by removing the the rows with mostly NA values and near-zero-variance (NZV) variables. Then, we will update our train_prelim and test_prelim in every process.

```
# Clean variables with NZV
nzv_variables <- nearZeroVar(train_prelim)
train_prelim <- train_prelim[,-nzv_variables]
test_prelim <- test_prelim[,-nzv_variables]

# Remove variables with mostly null values. We set 95% as our threshold
na_variables <- sapply(train_prelim, function(x) mean(is.na(x))) > 0.95
train_prelim <- train_prelim[,na_variables == FALSE]
test_prelim <- test_prelim[,na_variables == FALSE]

# After the cleaning process, we see that we are left with just 59 columns.
dim(train_prelim)
```

```
## [1] 13737    59
```

```
dim(test_prelim)
```

```
## [1] 5885   59
```

Looking at the first 5 columns of our train_prelim, we see that these are just identifier variables and will not be needed for our prediction. Thus, we drop these columns and we are now left with just 54 predictors.

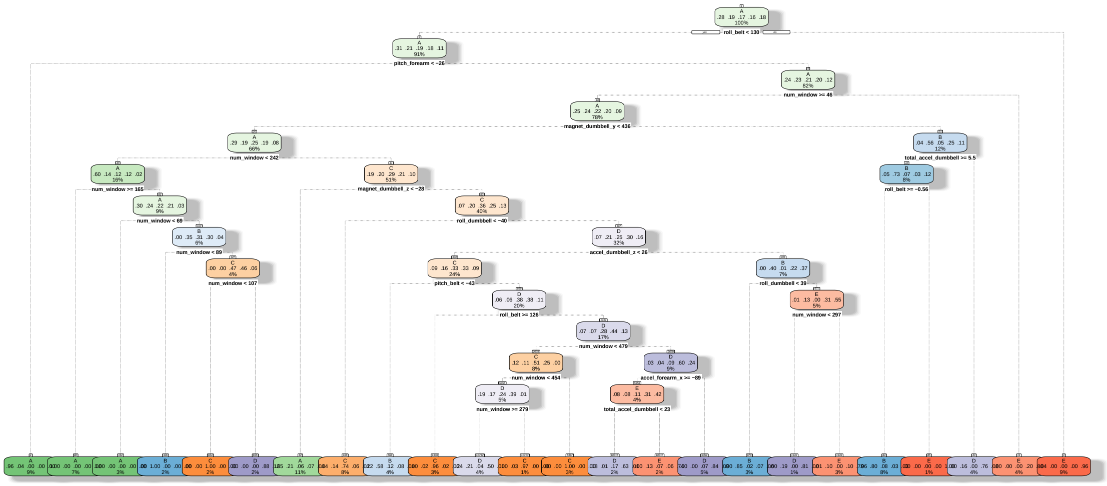
```
train_prelim <- train_prelim[,-(1:5)]
test_prelim <- test_prelim[,-(1:5)]
```

Prediction Models

In this project, we will build three models: Decision Tree, Random Forest model, and Generalized Boosted Model (GBM). We will train these models using the train_prelim and validate their accuracy using test_prelim. After the models have been built, we will choose the model with a higher accuracy and apply that to our test data set.

Decision Tree

```
library(rpart)
library(rattle)
set.seed(999)
DT_model <- rpart(classe~., data = train_prelim, method = "class")
fancyRpartPlot(DT_model)
```



Rattle 2023−Aug−06 19:12:51 keanafrancheskabautista

After we have built our model, we will apply it on our test_prelim

```
DT_predict <- predict(DT_model, test_prelim, type="class")
DT_conf <- confusionMatrix(DT_predict,factor(test_prelim$classe))
DT_conf
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction    A    B    C    D    E
##            A 1435  160   41   43   10
##            B  105  774   66   43   32
##            C   11   74  853   35    5
##            D   94   98   52  773   79
##          E   29   33   14   70  956
## 
## Overall Statistics
## 
##                Accuracy : 0.8141          
##                  95% CI : (0.8039, 0.824) 
##     No Information Rate : 0.2845          
##     P-Value [Acc > NIR] : < 2.2e-16       
## 
##                   Kappa : 0.765           
## 
##  Mcnemar's Test P-Value : 2.084e-14       
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8572   0.6795   0.8314   0.8019   0.8835
## Specificity            0.9397   0.9482   0.9743   0.9344   0.9696
## Pos Pred Value         0.8496   0.7588   0.8722   0.7053   0.8675
## Neg Pred Value         0.9430   0.9250   0.9647   0.9601   0.9737
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2438   0.1315   0.1449   0.1314   0.1624
## Detection Prevalence   0.2870   0.1733   0.1662   0.1862   0.1873
## Balanced Accuracy      0.8985   0.8139   0.9028   0.8681   0.9266
```

The results show a predictive accuracy of **74.66%** for our Decision Tree model. Next, we will see how a Random Forest model compares to our first model.

Random Forest Model

```
set.seed(999)
RF_model <- train(classe~., data = train_prelim, method = "rf",
                    trControl = trainControl(method = "repeatedcv",number = 5, repeats=2),
                  verbose=FALSE)
```

After we have built our model, we will apply it on our test_prelim

```
RF_predict <- predict(RF_model, test_prelim)
RF_conf <- confusionMatrix(RF_predict,factor(test_prelim$classe))
RF_conf
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    1    0    0    0
##          B    0 1137    2    0    0
##          C    0    1 1024    3    0
##          D    0    0    0  960    0
##          E    0    0    0    1 1082
## 
## Overall Statistics
##                                           
##                Accuracy : 0.9986          
##                  95% CI : (0.9973, 0.9994)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9983
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9982   0.9981   0.9959   1.0000
## Specificity            0.9998   0.9996   0.9992   1.0000   0.9998
## Pos Pred Value         0.9994   0.9982   0.9961   1.0000   0.9991
## Neg Pred Value         1.0000   0.9996   0.9996   0.9992   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1932   0.1740   0.1631   0.1839
## Detection Prevalence   0.2846   0.1935   0.1747   0.1631   0.1840
## Balanced Accuracy      0.9999   0.9989   0.9986   0.9979   0.9999
```

The results show a predictive accuracy of **99.81%** for our Random Forest model which was higher than the decision tree model. Finally, we will observe Generalized Boosted Modem compared to the first two models

Generalized Boosted Model

```
library(caret)
set.seed(999)
GB_model <- train(classe~., data = train_prelim, method = "gbm",
                  trControl = trainControl(method = "repeatedcv",number = 5, repeats=2),
                  verbose=FALSE)
```

After we have built our model, we will apply it on our test_prelim

```
GB_predict <- predict(GB_model, test_prelim)
GB_conf <- confusionMatrix(GB_predict,factor(test_prelim$classe))
GB_conf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1667    3    0    0    0
##          B    4 1127   11   10    7
##          C    0    8 1010   12    2
##          D    1    1    3  941   16
##          E    2    0    2    1 1057
##
## Overall Statistics
##
##                Accuracy : 0.9859
##                  95% CI : (0.9825, 0.9888)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9822
##
```

```
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9958   0.9895   0.9844   0.9761   0.9769
## Specificity            0.9993   0.9933   0.9955   0.9957   0.9990
## Pos Pred Value         0.9982   0.9724   0.9787   0.9782   0.9953
## Neg Pred Value         0.9983   0.9975   0.9967   0.9953   0.9948
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2833   0.1915   0.1716   0.1599   0.1796
## Detection Prevalence   0.2838   0.1969   0.1754   0.1635   0.1805
## Balanced Accuracy      0.9976   0.9914   0.9899   0.9859   0.9879
```

The results show a predictive accuracy of 98.69% for our Generalized Boosted Model which was lower than the random forest model.

Summary of Results and Choosing the Best Predictive Model

From our series of model analysis, we see that the decision tree model had the lowest predictive accuracy of **75.2%** among the three while both RF and GBM models had almost equal predictive accuracy with **99.81%** and **98.69%** respectively.

Although both RF and GMB can be used on our quiz dataset, we will proceed with the random forest model. Let's now apply our final model to the test data.

```r
final_predict <- as.data.frame(predict(RF_model, test_data))
final_predict
```

```
##    predict(RF_model, test_data)
## 1                             B
## 2                             A
## 3                             B
## 4                             A
## 5                             A
## 6                             E
## 7                             D
## 8                             B
## 9                             A
## 10                            A
## 11                            B
## 12                            C
## 13                            B
## 14                            A
## 15                            E
## 16                            E
## 17                            A
## 18                            B
## 19                            B
## 20                            B
```