Practiacal Machine Learning Prediction Assignment

Synopsis

This assignment is to build a prediction model to apply in the prediction quiz. Two models will be built, a Decision Tree model and a Random Forest model, and evaluated. The model with the better accuracy in predicting on a validation set will be selected.

Load R Packeges

```
library(caret)
```

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

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

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(rpart)
library(rattle)
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

Load the Data

```
Training.Data <- read.csv("pml-training.csv")
Testing.Data <- read.csv("pml-testing.csv")
```

Clean the Data

The data set is very large and contains many variables that are not useful as predictors. The first five variables consist of a row number (X) the users names and three time stamps. These are excluded because they do not relate to any of the measurements that are used to make the prediction.

```
Training.Data <- Training.Data[,-(1:5)]
Testing.Data <- Testing.Data[,-(1:5)]
```

Next, variables that have very little variance are removed. Near zero variance variables use up a lot of computational time and offer little additional predictive power.

```
Zero.Var <- nearZeroVar(Training.Data)
Training.Data <- Training.Data[,-Zero.Var]
Testing.Data <- Testing.Data[,-Zero.Var]
```

Finally, there are many variables that contain a lot of NA values. All variables that contain more than five NAs are removed.

```
NAs <- colSums(is.na(Testing.Data))>5
Training.Data <- Training.Data[, NAs == FALSE]
Testing.Data <- Testing.Data[, NAs == FALSE]
```

Partition the Training Data into a Training and a Validation Set

The Training.Data data set is split into a Training data set and a Validation data set to perform cross-validation on after the models have been built.

```
set.seed(1316)
inTrain <- createDataPartition(Training.Data$classe, p = 0.6, list = FALSE)
Training <- Training.Data[inTrain,]
Validation <- Training.Data[-inTrain,]
```

Training the Models

Random Forests Model

The first model built is a random forest model. Also shown are the 20 most important variables in the

random forest model.

```
set.seed(1316)
model.RF <- train(classe ~ ., data = Training, method = "rf", trControl=trainCo
ntrol(method = "cv", number = 9), na.action = na.omit)
model.RF
```

```
## Random Forest
##
## 11776 samples
##    53 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (9 fold)
## Summary of sample sizes: 10468, 10465, 10468, 10469, 10467, 10468, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9932906  0.9915127
##   27    0.9960082  0.9949506
##   53    0.9931223  0.9912995
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

```
varImp(model.RF)
```

```
## rf variable importance
##
##   only 20 most important variables shown (out of 53)
##
##                       Overall
## num_window           100.000
## roll_belt             66.168
## pitch_forearm         41.846
## yaw_belt              31.233
## magnet_dumbbell_z     30.319
## magnet_dumbbell_y     29.417
## pitch_belt            28.975
## roll_forearm          24.100
## accel_dumbbell_y      14.380
## accel_forearm_x       10.816
## magnet_dumbbell_x     10.679
## roll_dumbbell         10.663
## total_accel_dumbbell   9.809
## accel_belt_z           9.546
## accel_dumbbell_z       8.088
## magnet_belt_z          7.578
## magnet_belt_y          7.412
## magnet_forearm_z       7.165
## magnet_belt_x          6.283
## roll_arm               5.498
```

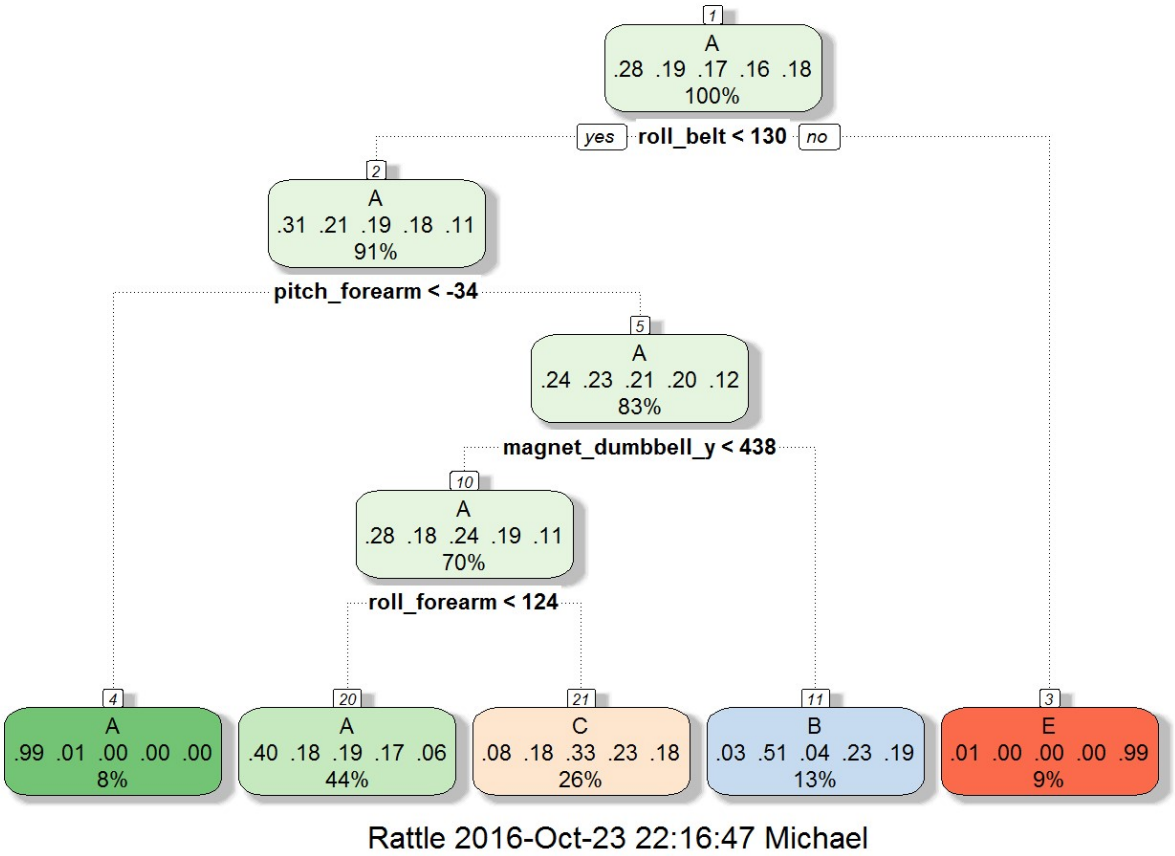
rpart Model

The second model built is a decision tree model. Also shown is a plot of the decision tree.

```
set.seed(1316)
model.rpart <- train(classe ~ . , data = Training, method = "rpart", na.action
= na.omit)
model.rpart
```

```
## CART
##
## 11776 samples
##    53 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 11776, 11776, 11776, 11776, 11776, 11776, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.04028239  0.5358964  0.40366812
##   0.06035437  0.4142367  0.20607521
##   0.11699098  0.3352688  0.07911218
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was cp = 0.04028239.
```

```
fancyRpartPlot(model.rpart$finalModel)
```



Cross-validation: Testing the Models on the Validation Set.

Both models are tested against the Validation data set to see which performs better.

Random Forest Model

```
Validate.RF <- predict(model.RF, Validation)
confusionMatrix(Validation$classe, Validate.RF)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2230    1    0    0    1
##          B    3 1515    0    0    0
##          C    0    2 1366    0    0
##          D    0    0    7 1279    0
##          E    0    0    0    1 1441
##
## Overall Statistics
##
##                Accuracy : 0.9981
##                  95% CI : (0.9968, 0.9989)
##     No Information Rate : 0.2846
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9976
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9987   0.9980   0.9949   0.9992   0.9993
## Specificity            0.9996   0.9995   0.9997   0.9989   0.9998
## Pos Pred Value         0.9991   0.9980   0.9985   0.9946   0.9993
## Neg Pred Value         0.9995   0.9995   0.9989   0.9998   0.9998
## Prevalence             0.2846   0.1935   0.1750   0.1631   0.1838
## Detection Rate         0.2842   0.1931   0.1741   0.1630   0.1837
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9992   0.9988   0.9973   0.9991   0.9996
```

The random forest model was 99.8% accurate in the cross-validation test. This means it had an out of samle error rate of 0.2%

Decision Tree Model

```
Validate.Rpart <- predict(model.rpart, Validation)
confusionMatrix(Validation$classe, Validate.Rpart)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2029   41  157    0    5
##          B  643  505  370    0    0
##          C  622   47  699    0    0
##          D  573  220  493    0    0
##          E  212  198  396    0  636
##
## Overall Statistics
##
##                Accuracy : 0.4931
##                  95% CI : (0.482, 0.5042)
##     No Information Rate : 0.5199
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3375
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.4974  0.49951  0.33050       NA  0.99220
## Specificity            0.9461  0.85179  0.88327   0.8361  0.88813
## Pos Pred Value         0.9091  0.33267  0.51096       NA  0.44105
## Neg Pred Value         0.6348  0.92004  0.78141       NA  0.99922
## Prevalence             0.5199  0.12886  0.26956   0.0000  0.08170
## Detection Rate         0.2586  0.06436  0.08909   0.0000  0.08106
## Detection Prevalence   0.2845  0.19347  0.17436   0.1639  0.18379
## Balanced Accuracy      0.7218  0.67565  0.60688       NA  0.94017
```

The decision tree model was 49.3% accurate in the cross-validation test. This means it had an out of samle error rate of 50.7%

Model Selection

Based on the outcome of the validation tests, the Random Forests prediction model as it was 99.8% accurate in the validation test compared to the Decision Tree model at 49.3%

Predicting on the Test data

Use the random forest model to predict the exercise type based on the variables in the Testing.Data data set.

```
Testing.RF <- predict(model.RF, Testing.Data)
matrix(Testing.RF, 20, 1)
```

```
##       [,1]
##  [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"
```