Practical Machine Learning Assignment

Summary

People regularly quantify how much activity they do, but they rarely quantify how well they do it. In this project, the goal is to analyze data from accelerometers on the belt, forearm, arm, and dumbell of six participants. They were asked to perform barbell lifts correctly and incorrectly in five different ways. ar

The goal of this machine learning exercise is to predict the manner in which the participants did the exercise, that is to predict the "classe" variable found in the training dataset. The prediction model will then used to predict 20 different test cases.

Data Processing and Analysis

Loading the required libraries and reading in the training and testing datasets, assign missing values to values that are 'NA' or blank.

```
set.seed(1)
```

```
datTraining<-read.csv("pml-training.csv", header=TRUE, na.strings=c("NA", ""))
datTesting<-read.csv("pml-testing.csv", header=TRUE, na.string=c("NA", ""))
```

Data Cleaning

Excluding data with "NA" and delete additional unnecessary columns

```
library(caret)
```

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

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

```
datTrainNoNA<-datTraining[, apply(datTraining, 2, function(x) !any(is.na(x)))]
dim(datTrainNoNA)
```

```
## [1] 19622    60
```

```
datTrainingClean<-datTrainNoNA[,-c(1:8)]
dim(datTrainingClean)
```

```
## [1] 19622    52
```

```
datTrainNzv <- nearZeroVar(datTrainingClean, saveMetrics=TRUE)
datTrainingCleanNew <- datTrainingClean[,datTrainNzv$nzv==FALSE]
dim(datTrainingCleanNew)
```

```
## [1] 19622    52
```

Data Partitioning and Prediction Process

Split the clean training dataset into a training dataset (70% of the observations) and a validation dataset (30% of the observations). This validation dataset is use to perform cross validation when developing the model.

```
inTrainingData<-createDataPartition(y=datTrainingCleanNew$classe, p=0.70,list=FALSE)
myTrainData <- datTrainingCleanNew[inTrainingData,]
dim(myTrainData)
```

```
## [1] 13737    52
```

```
myTestData <- datTrainingClean[-inTrainingData,]
dim(myTestData)
```

```
## [1] 5885   52
```

Prediction Model

Prediction with Random Forests

```
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
```

```
fitControl <- trainControl(method="cv", number=5, allowParallel=T, verbose=T)
rfModel<-train(classe~.,data=myTrainData, method="rf", trControl=fitControl, verbose=F)
```

```
## + Fold1: mtry= 2
## - Fold1: mtry= 2
## + Fold1: mtry=26
## - Fold1: mtry=26
## + Fold1: mtry=51
## - Fold1: mtry=51
## + Fold2: mtry= 2
## - Fold2: mtry= 2
## + Fold2: mtry=26
## - Fold2: mtry=26
## + Fold2: mtry=51
## - Fold2: mtry=51
## + Fold3: mtry= 2
## - Fold3: mtry= 2
## + Fold3: mtry=26
## - Fold3: mtry=26
## + Fold3: mtry=51
## - Fold3: mtry=51
## + Fold4: mtry= 2
## - Fold4: mtry= 2
## + Fold4: mtry=26
## - Fold4: mtry=26
## + Fold4: mtry=51
## - Fold4: mtry=51
## + Fold5: mtry= 2
## - Fold5: mtry= 2
## + Fold5: mtry=26
## - Fold5: mtry=26
## + Fold5: mtry=51
## - Fold5: mtry=51
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 26 on full training set
```

```
print(rfModel)
```

```
## Random Forest
##
## 13737 samples
##    51 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 10988, 10990, 10990, 10990, 10990
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9904634  0.9879353
##   26    0.9906818  0.9882122
##   51    0.9866051  0.9830539
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 26.
```

Evaluate the Model

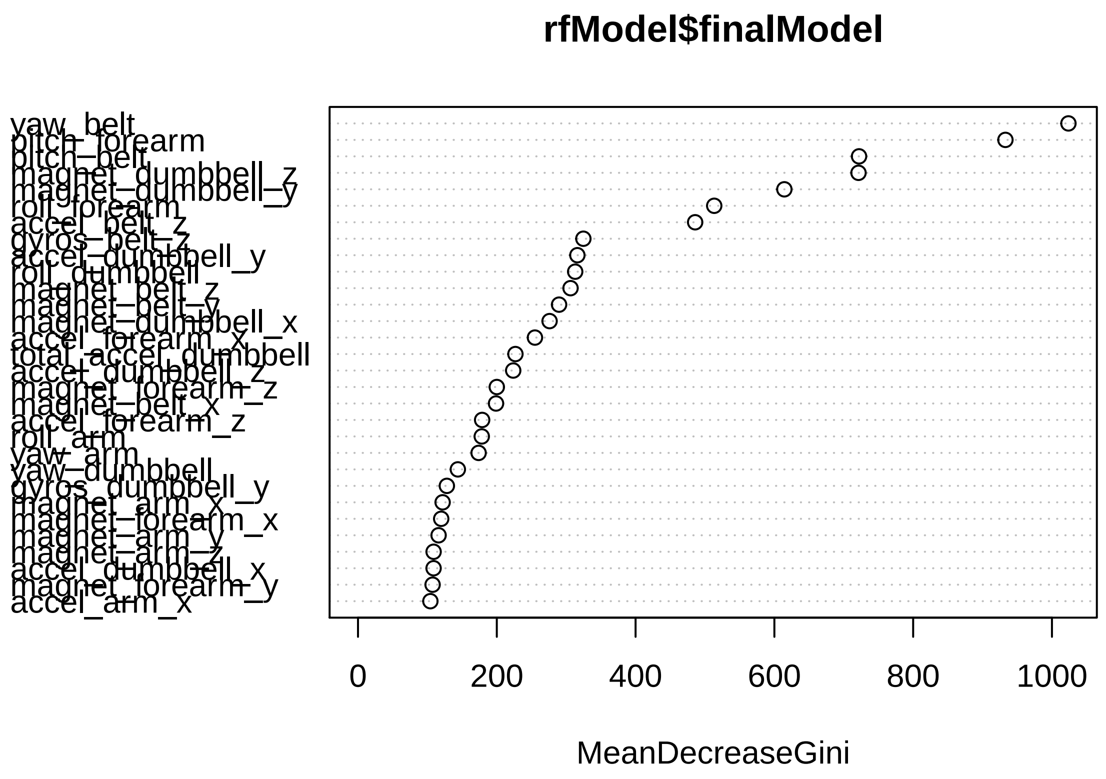
Using confusionMatrix to evaluate the model

```
predrf<-predict(rfModel, myTestData)
confusionMatrix(predrf, myTestData$classe)
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    7    0    0    0
##          B    1 1130    8    0    0
##          C    1    2 1016   12    1
##          D    0    0    2  952    2
##          E    0    0    0    0 1079
##
## Overall Statistics
##
##                Accuracy : 0.9939
##                  95% CI : (0.9915, 0.9957)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9923
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9988   0.9921   0.9903   0.9876   0.9972
## Specificity            0.9983   0.9981   0.9967   0.9992   1.0000
## Pos Pred Value         0.9958   0.9921   0.9845   0.9958   1.0000
## Neg Pred Value         0.9995   0.9981   0.9979   0.9976   0.9994
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2841   0.1920   0.1726   0.1618   0.1833
## Detection Prevalence   0.2853   0.1935   0.1754   0.1624   0.1833
## Balanced Accuracy      0.9986   0.9951   0.9935   0.9934   0.9986
```

```r
varImpPlot(rfModel$finalModel)
```



X-axis is the degree of importance axis. This plot shows the degree of importance of each of the principal components

Result of the prediction to test dataset

The accuracy is 99%, thus the predicted accuracy for the out of sample error is 1%. Random Forests is chosen because it produce better results.

```
predictpmlTesting<-predict(rfModel, datTesting)

#Output for the prediction of the 20 cases provided
predictpmlTesting
```

```
##  [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
```

20/20 cases we correctly predicted