Practical Machine Learning Project

Abstract

Using Human Activity Recognition (HAR) data that was collected from sensors on the human body while a person performs an exercise, a prediction model was constructed to determine what activity was conducted based off the sensor data provided.

After pre-processing the training data and reducing the data to useable features, several models were fit to the training data and their in-sample accuracy was assessed. The final model that was selected was a Random Forest model (method = rf) that had a 99.5% in-sample accuracy rate and provided a cross-validation of 20 out of 20 correctly matched predictions against the testing dataset. The other model that was testing and produced the same cross-validation accuracy of 20 out of 20 prediction match rate was the Generalized Boosted Model (method = gbm) and had a 98.6% in sample accuracy rate. The analysis below documents the steps take to achieve the “best fit” Random Forest model.

More information on the data and other HAR studies can be found here: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har)

```
library(caret)
library(randomForest)
library(ggplot2)
library(e1071)
library(rpart)
library(MASS)
library(markdown)
library(knitr)
```

Read in the training and testing data. Because the data was already separated into two separate training and testing files there was no need to partition the data.

```
# Set the Working directory for the project
setwd("C:/Users/Mark Maxwell/Desktop/Coursera/Data Science Specialization Track/08. Practical M
achine Learning/Course Project/Data")

# Read in the testing dataset
testing <- read.csv("pml-testing.csv",header = TRUE)

# Read in the training dataset
training <- read.csv("pml-training.csv",header = TRUE)
```

Preprocess the training data.

Processing the dataset included removing variables that had no predictive power such as the name of the person performing the exercise and the timestamp that the exercise was performed. Additionally, the dataset contained a number of variables that were mostly NAs or were persisted in the training dataset as blanks.

These variables were removed from the dataset before any models were fit to remove noise and decrease the time needed to compute each model.

```
# Remove variables form the training dataset that have no predictive power
training <- droplevels(training[,!names(training) %in% c("X","raw_timestamp_part_1","raw_timest
amp_part_2","cvtd_timestamp","user_name")])

# Reomve NAs form the training dataset
training <- droplevels(training[,colSums(is.na(training)) == 0])

# Reomve all the columns that are mostly empty cells
training <- droplevels(training[,colSums(training=="") == 0])
```

Testing several models.

Several models were fit then their predictions were reviewed and checked for their in sample accuracy. Below is the code for the process used to test each model but only the final Random Forest model is evaluated below in this document.

```
# Linear Discriminate Analysis Model
LDAmodelFit <- train(classe ~ ., method = "lda", data = training)
predict(LDAmodelFit,testing)
LDAmodelFit

# Naive Bayes Model
NBmodelFit <- train(classe ~ ., method = "nb", data = training)
predict(NBmodelFit,testing)
NBmodelFit

# General Boosted Model
GBMmodelFit <- train(classe ~ ., method = "gbm", verbose = FALSE, data = training)
predict(GBMmodelFit,testing)
GBMmodelFit

# Classification Tree
RPARTmodelFit <- train(classe ~ ., method = "rpart", data = training)
predict(RPARTmodelFit,testing)
RPARTmodelFit

# Random Forest Model
RFmodelFit <- train(classe ~ ., method = "rf", data = training)
predict(RFmodelFit,testing)
RFmodelFit
```

Final Model Selection.

The final model that was selected for use was the Random Forest model that had a 99.5% in sample accuracy rate and provided a cross-validated match rate of 20 out of 20 correctly matched predictions against the testing dataset.

*The model results below are printed out in the markdown file so they would not have to be recomputed every time the document is knit to HTML.

```
RFmodelFit <- train(classe ~ ., method="rf", data=training)
```

```
predict(RFmodelFit,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
```

```
RFmodelFit
```

```
## Random Forest
##
## 19622 samples
##    54 predictors
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 19622, 19622, 19622, 19622, 19622, 19622, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
##   2     1         1      0.001        0.001
##   30    1         1      6e-04        8e-04
##   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 = 28.
```

```
RFmodelFit$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 28
##
##         OOB estimate of  error rate: 0.14%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 5578    1    0    0    1   0.0003584
## B    6 3788    2    1    0   0.0023703
## C    0    4 3417    1    0   0.0014611
## D    0    0    8 3207    1   0.0027985
## E    0    0    0    3 3604   0.0008317
```