Practical Machine Learning: Assignment

Leslie Bogdan

5 February 2018

Data source and Back ground

Background

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 project, your goal will be to use 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. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har).

Aim

Generate a prediction model (with reference to the supplied training data sets) which will then be used to predict on a new (unknown) data set (will then be quized on the models performance).

Libraries used

- library(ggplot2)
- library(caret)
- library(rpart)
- library(rpart.plot)
- library(rattle)
- library(randomForest)
- library(corrplot)

Loading the Data

Download the two data sets (training and testing) csv files

load them into dataframes

We have been asked to set.seed = ‘12345’

```
set.seed(12345)
```

Make sure you are pointing at the correct working directory which contains the csv files needed (as outlined in the datasets section above)

```
testing<-read.csv("pml-testing.csv", header=TRUE, sep=",")
training<-read.csv("pml-training.csv", header=TRUE, sep=",")
```

Cross Validation

Create the training and test sets (from the training dataframe) to be used against future prediction approaches.

```
train_partition  <- createDataPartition(training$classe, p=0.7, list=FALSE)
train_set <- training[train_partition, ]
test_set  <- training[-train_partition, ]
```

Data Cleaning up

Looking at the current state of both the ‘original_train’ there are many variables which seem to have many values which are close to zero. We will now go and clean these up.

The dataset also seems to have allot of varibales which have allot of missing values, we also clean them up in this step.

Finally, we remove the variables which are related to the identification of subjects.

```
# Variables where majority of values are near 0 removed

near_zero_value <- nearZeroVar(train_set)
train_set <- train_set[, -near_zero_value]
test_set  <- test_set[, -near_zero_value]

# Variables where majority are NA values removed

NA_remove    <- sapply(train_set, function(x) mean(is.na(x))) > 0.9
train_set <- train_set[, NA_remove==FALSE]
test_set  <- test_set[, NA_remove==FALSE]


# Identifier columns to be removed

train_set <- train_set[, -(1:5)]
test_set  <- test_set[, -(1:5)]
```

Prediction Model Generation

We will now use the following two approaches to generate prediction models within the training partition portion of the training data.

- Random Forest
- Decision Trees

Based on the above prediction model performance against the test partition portion of the training data set (based on the 'Accuracy' measure (highest is better)), we will then apply the selected model to predict the quiz test cases .

```
# Random Forest Approach

control_rf <- trainControl(method="cv", number=3, verboseIter=FALSE)
model_rf <- train(classe ~ ., data=train_set, method="rf",
                          trControl=control_rf)
model_rf$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: 27
##
##         OOB estimate of  error rate: 0.21%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    1    0    0    1 0.0005120328
## B    6 2650    1    1    0 0.0030097818
## C    0    4 2392    0    0 0.0016694491
## D    0    0    7 2245    0 0.0031083481
## E    0    2    0    6 2517 0.0031683168
```

```
        # Decision Trees Approach

model_dt <- rpart(classe ~ ., data=train_set, method="class")
```

Prediction Model Application and Evaluation

Apply them to the Test portion of the training data set we partioned earlier.

```
# Apply Random Forest Approach

pred_rf <- predict(model_rf, newdata=test_set)
conf_model_rf <- confusionMatrix(pred_rf, test_set$classe)
conf_model_rf
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1133    3    0    0
##          C    0    1 1023    8    0
##          D    0    0    0  956    4
##          E    0    0    0    0 1078
## 
## Overall Statistics
## 
##                Accuracy : 0.9964
##                  95% CI : (0.9946, 0.9978)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9955
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9971   0.9917   0.9963
## Specificity            0.9988   0.9994   0.9981   0.9992   1.0000
## Pos Pred Value         0.9970   0.9974   0.9913   0.9958   1.0000
## Neg Pred Value         1.0000   0.9987   0.9994   0.9984   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1925   0.1738   0.1624   0.1832
## Detection Prevalence   0.2853   0.1930   0.1754   0.1631   0.1832
## Balanced Accuracy      0.9994   0.9971   0.9976   0.9954   0.9982
```

```
rf <- conf_model_rf$overall['Accuracy']
rf_out<-1-rf

# Apply the Decision Tree Approach

pred_dt <- predict(model_dt, newdata=test_set, type="class")
conf_model_dt <- confusionMatrix(pred_dt, test_set$classe)
conf_model_dt
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 1530  269   51   79   16
##          B   35  575   31   25   68
##          C   17   73  743   68   84
##          D   39  146  130  702  128
##          E   53   76   71   90  786
## 
## Overall Statistics
## 
##                Accuracy : 0.7368
##                  95% CI : (0.7253, 0.748)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.6656
##  Mcnemar's Test P-Value : < 2.2e-16
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9140  0.50483   0.7242   0.7282   0.7264
## Specificity            0.9014  0.96650   0.9502   0.9100   0.9396
## Pos Pred Value         0.7866  0.78338   0.7543   0.6131   0.7305
## Neg Pred Value         0.9635  0.89051   0.9422   0.9447   0.9384
## Prevalence             0.2845  0.19354   0.1743   0.1638   0.1839
## Detection Rate         0.2600  0.09771   0.1263   0.1193   0.1336
## Detection Prevalence   0.3305  0.12472   0.1674   0.1946   0.1828
## Balanced Accuracy      0.9077  0.73566   0.8372   0.8191   0.8330
```

```
dt<-conf_model_dt$overall['Accuracy']
dt_out<-1-dt
```

Results

In examining the ‘Accuracy’ measure produced from both prediction models, we get the following

- Random Forest @ 0.9964316
- Decision Trees @ 0.7367884

Based on the accuracy results, we select the ‘Random Forest’ model.

Out of sample errors

The Random Forest approach had an ‘out of sample error rate’Accuracy’ value of 0.9964316 and out of sample error of 0.0035684, while the decision tree approach had a ‘Accuracy’ value of 0.7367884 and out of sample error rate of 0.2632116.

Hence, the random forest approach has been selected to be used on the test data set due it having the higher accuracy value.

Application to the Test (Quiz) set

We now use our selected model to predict across the 'testing' (quiz) data set.

head on first 20 records of the predictions produced (needed for the quiz).

```
pred_final <- predict(model_rf, newdata=testing)
head(pred_final,20)
```

```
##  [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
```