Practical Machine Learning - Programming Assignment

Rajesh Ekkaladevi

Mar 19, 2020

Executive Summary

The goal of the exercise is to review Human Activity Recognition (HAR) data captured and use Machine Learning techniques to model the data and predict results based on test data.

More details about the data is in the "Background Info" section.

First a glimpse of the input data is made. Then using exploratory data analysis base set of features selected for further review. Later feature selection is performed and then multiple machine learning algorithms are applied and compared for accuracy. Finally the best model is used to predict results on the test data.

The goal is to predict the human activity performed based on data received from each of the sensor during the activity. It is a classification problem.

The findings are among the 3 models tried (MARS, Random Forests, Decision Trees) the model with best accuracy is obtained from Random Forests algorithm.

Background Info

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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project is available at: pml-training

The test data is available at: pml-testing

Reference

The data for this project provided by: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.

Data Processing

Setting seed for reproducibility

```
set.seed(54321)
```

Reading input data

```
training <- read_csv(url('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'))
testing <- read_csv(url('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'))
```

The input data has the following dimension:

```
training %>% dim()
```

```
## [1] 19622   160
```

Partition training data into train_set and test_set

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

Preprocessing (Feature Filtering, Extraction and Engineering)

Feature Filtering, Extraction and Engineering

From all data sets removing aggregated measures and keeping facts of importance only. Timestamp is ignored as the input data is captured for few hours only and the goal is to predict the activity rather forecast. Also removed other obvious info like new_window, num_window and X1.

```
train_set_fix <- train_set %>% filter(new_window=='no') %>%
  select(-starts_with(c('avg','var', 'std', 'min', 'max', 'amplitude', 'kurtosis', 'skewness',
  'user_name', 'raw_timestamp_part_1', 'raw_timestamp_part_2', 'cvtd_timestamp', 'new_window',
  'num_window', 'X1'))) %>%
  mutate(classe = factor(classe))

test_set_fix <- test_set %>% filter(new_window=='no') %>%
  select(-starts_with(c('avg','var', 'std', 'min', 'max', 'amplitude', 'kurtosis', 'skewness',
  'user_name', 'raw_timestamp_part_1', 'raw_timestamp_part_2', 'cvtd_timestamp', 'new_window',
  'num_window', 'X1'))) %>%
  mutate(classe = factor(classe))

testing_fix <- testing %>% filter(new_window=='no') %>%
  select(-starts_with(c('avg','var', 'std', 'min', 'max', 'amplitude', 'kurtosis', 'skewness',
  'user_name', 'raw_timestamp_part_1', 'raw_timestamp_part_2', 'cvtd_timestamp', 'new_window',
  'num_window', 'X1')))
```

Check for missing data

Checking to see no variable is missing data:

```
train_set_fix %>% anyNA()
```

```
## [1] FALSE
```

The input data after filtering un-necessary variables has the following dimension:

```
train_set_fix %>% dim()
```

```
## [1] 13449    53
```

```
test_set_fix %>% dim()
```

```
## [1] 5767   53
```

```
testing_fix %>% dim()
```

```
## [1] 20 53
```

Scaling Features

Transforming (standardizing) variable values between 0 and 1 except the outcome variable (classe). This not only avoids unequal weightage to different variables but also helps compare relative effect of each variable with others from different scale.

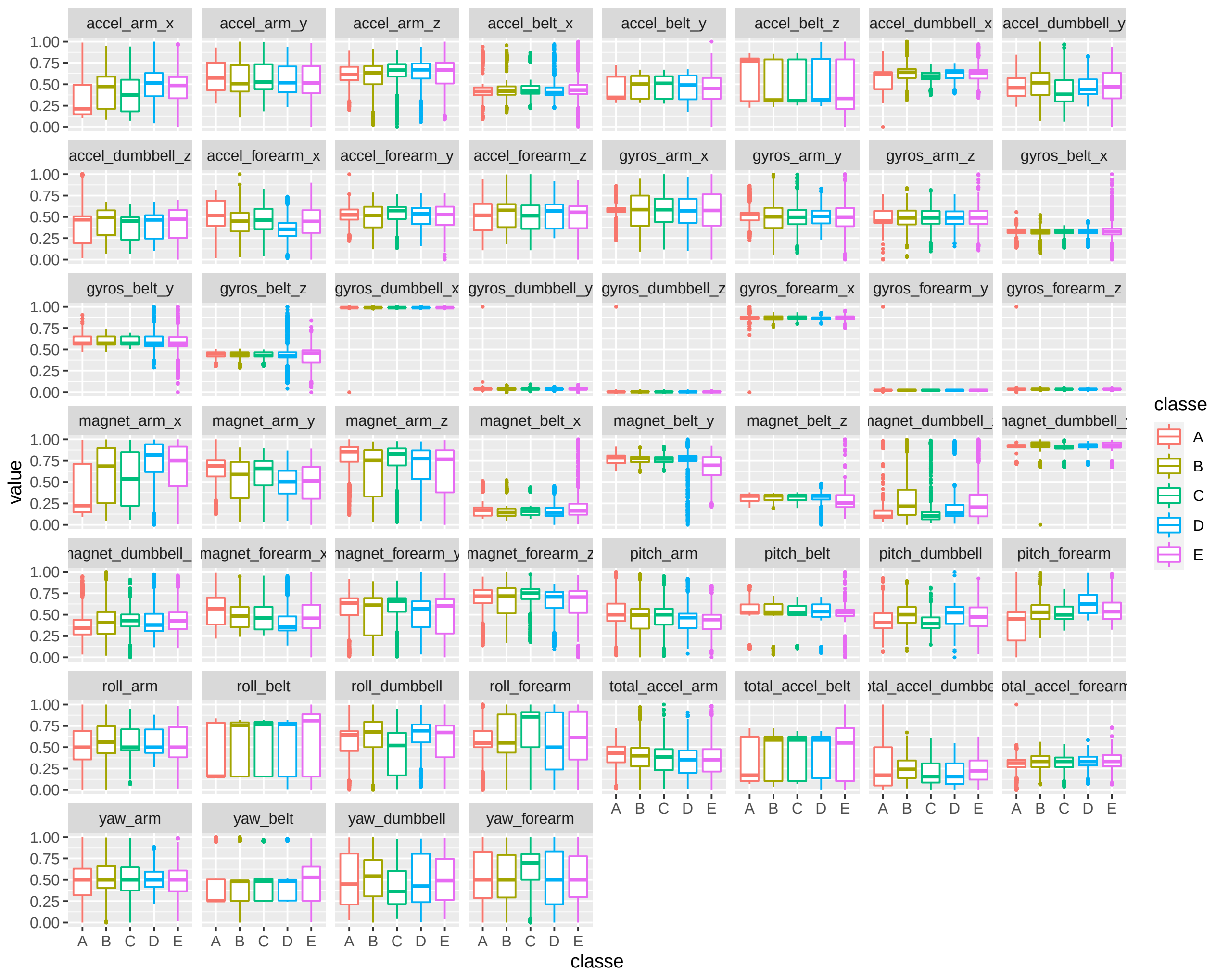
```
trans_fit <- preProcess(train_set_fix, method='range')

train_set_fix_trans <- predict(trans_fit, newdata=train_set_fix)
train_set_fix_trans$classe <- train_set_fix$classe

test_set_fix_trans <- predict(trans_fit, newdata=test_set_fix)
test_set_fix_trans$classe <- test_set_fix$classe

testing_fix_trans <- predict(trans_fit, newdata=testing_fix)
```

Exploratory Data Analysis

Pair plot with outcome (classe)

```
train_set_fix_trans_plot <- train_set_fix_trans %>% gather(variable, value, -classe)
```

```
ggplot(train_set_fix_trans_plot, aes(classe, value, color=classe)) +
  geom_boxplot(outlier.size=0.4) +
  facet_wrap(vars(variable))
```



Model Fit

Cross validation setup for training

```
train_control <- trainControl(method = "repeatedcv", number = 5, repeats = 1, verbose=FALSE)
```

Fit multiple models and check accuracy

Multiple Adaptive Regression Splines (MARS)

```
fit_mars <- train(classe ~ ., data=train_set_fix_trans, method='earth', trControl=train_control)
fit_mars
```

```
## Multivariate Adaptive Regression Spline
##
## 13449 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 1 times)
## Summary of sample sizes: 10758, 10760, 10760, 10759, 10759
## Resampling results across tuning parameters:
##
##   nprune  Accuracy   Kappa
##    2      0.3493947  0.1222386
##   16      0.7100886  0.6324734
##   30      0.7630345  0.6998744
##
## Tuning parameter 'degree' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were nprune = 30 and degree = 1.
```

Random Forests

```
fit_rf <- train(classe ~ ., data=train_set_fix_trans, method='rf', trControl=train_control)
fit_rf
```

```
## Random Forest
##
## 13449 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 1 times)
## Summary of sample sizes: 10759, 10759, 10759, 10760, 10759
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9906309  0.9881457
##   27    0.9910029  0.9886172
##   52    0.9863187  0.9826916
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 27.
```

Decision Trees

```
fit_rpart <- train(classe ~ ., data=train_set_fix_trans, method='rpart', trControl=train_control)
fit_rpart
```

```
## CART
##
## 13449 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 1 times)
## Summary of sample sizes: 10758, 10759, 10759, 10760, 10760
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.03576627  0.5134953  0.36863860
##   0.06023428  0.4430763  0.25305629
##   0.11665627  0.3171212  0.04916072
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.03576627.
```

Compare models for best accuracy

```
fit_compare <- resamples(list(MARS=fit_mars, RF=fit_rf, DT=fit_rpart))
fit_compare %>% summary
```

```
##
## Call:
## summary.resamples(object = .)
##
## Models: MARS, RF, DT
## Number of resamples: 5
##
## Accuracy
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## MARS 0.7454478 0.7486054 0.7539033 0.7630345 0.7565056 0.8107103    0
## RF   0.9877323 0.9895872 0.9910781 0.9910029 0.9933086 0.9933086    0
## DT   0.4868079 0.4990703 0.4998141 0.5134953 0.5144981 0.5672862    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## MARS 0.6778104 0.6813829 0.6881430 0.6998744 0.6916269 0.7604088    0
## RF   0.9844806 0.9868240 0.9887117 0.9886172 0.9915342 0.9915354    0
## DT   0.3302432 0.3459477 0.3466793 0.3686386 0.3644468 0.4558761    0
```

Use best model to predict on internal validation set

```
predicted_rf <- predict(fit_rf, newdata=test_set_fix_trans)

confusionMatrix(reference=test_set_fix$classe, data=predicted_rf)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1638    7    0    0    0
##          B    2 1104    3    1    0
##          C    0    3 1001    5    2
##          D    0    0    2  941    3
##          E    0    0    1    0 1054
##
## Overall Statistics
##
##                Accuracy : 0.995
##                  95% CI : (0.9928, 0.9966)
##     No Information Rate : 0.2844
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9936
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9988   0.9910   0.9940   0.9937   0.9953
## Specificity            0.9983   0.9987   0.9979   0.9990   0.9998
## Pos Pred Value         0.9957   0.9946   0.9901   0.9947   0.9991
## Neg Pred Value         0.9995   0.9979   0.9987   0.9988   0.9989
## Prevalence             0.2844   0.1932   0.1746   0.1642   0.1836
## Detection Rate         0.2840   0.1914   0.1736   0.1632   0.1828
## Detection Prevalence   0.2852   0.1925   0.1753   0.1640   0.1829
## Balanced Accuracy      0.9985   0.9949   0.9960   0.9963   0.9975
```

Test (external data)

Use best model to predict on final test data

```
predicted_rf <- predict(fit_rf, newdata=testing_fix_trans)
predicted_rf
```

```
##  [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
```